VŠB – Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

BAKALÁŘSKÁ PRÁCE

2011 Jakub Hlavica

VŠB – Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

Katedra měřicí a řídicí techniky

Klasifikace komplexních metabolických pacientských dat

Classification of Complex Metabolic Pacient‘s Data

2011 Jakub Hlavica

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra měřicí a řídicí techniky

Zadání bakalářské práce

Student: **Jakub Hlavica**

Studijní program: B2649 Elektrotechnika

Studijní obor: 2601R004 Měřicí a řídicí technika

Téma: Klasifikace komplexních metabolických pacientských dat
Classification of Complex Metabolic Patient's Data

Zásady pro vypracování:

1. Měření metabolických dat.
2. Databázové prostředky pro medicínská data.
3. Návrh databáze pro klasifikaci metabolických dat.
4. Statistické zpracování metabolický parametrů.
5. Algoritmy klasifikace dat.
6. Realizace SW pro klasifikaci metabolických dat.
7. Testování pro klasifikaci a zhodnocení výsledků.

Seznam doporučené odborné literatury:

1. PENHAKER, M. - IMRAMOVSKÝ, M. - TIEFENBACH, P.*Lékařské diagnostické přístroje:učební texty* . 1. vyd. Ostrava: VŠB - Technická univerzita Ostrava, 2004. 320 s. ISBN 80-248-0751-3.
2. MOHYLOVÁ, J. - KRAJČA, V. *Zpracování signálů v lékařství.* [CD-ROM]. Žilinská universita, 2005. ISBN 80-8070-341-8.
3. BUREK, D. G. *Biosensors : theory and applications*. Lancaster(USA): Technomic, 1993. 232 s. ISBN 0-87762-975-7.
4. BRONZINO, J. D., et al. *The biomedical engineering handbook.* Boca Raton(USA): CRC Press, 1995. 1656 s. ISBN 084930461X.
5. WEBSTER, J.-G. *Medical instrumentation: application and design.* Hoboken (USA): Wiley, 1998. 691 s. ISBN 0-471-15368-0.
6. MARTINÍK, K. *Obezita, nadváha.* 3. vyd. Hradec Králové: Garamon s.r.o., 2008. 151 s. ISBN 978-80-86472-37-9.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Michal Prauzek**

Datum zadání: 19.11.2010

Datum odevzdání: 06.05.2011

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * OSTRAVA *

doc. Ing. Jiří Koziorek, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně a uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……..…….....

V Ostravě, dne 6. 5. 2011 *Jakub Hlavica*

Poděkování

Chtěl bych tímto poděkovat vedoucímu mé bakalářské práce Ing. Michalovi Prauzkovi za trpělivý a ochotný přístup během konzultací týkajících se vypracování této práce a za poskytnutou inspiraci při řešení komplexních problémů spojených s návrhem a vývojem software pro tvorbu databáze a klasifikace dat. Dále bych chtěl poděkovat Ing. Martinu Černému za cenné rady z oblasti programování v prostředí MATLAB.

Abstrakt

Cílem této bakalářské práce je vyvinout v programovacím prostředí MATLAB software, který je schopen klasifikovat velké množství komplexních metabolických pacientských dat získaných v průběhu klinických vyšetření pacientů nemocnice v Hradci Králové v letech 2007 až 2010. Program je založen na metodách umělé inteligence a jeho jádro je tvořeno neuronovou sítí s architekturou self-organizing map. Aplikace dokáže zpracovat vícerozměrná vstupní data a klasifikovat je do skupin s podobnými parametry, tzv. clustery. Na základě grafických a numerických výstupů z programu pak mohou lékaři určit, které metabolické parametry navzájem souvisí a které vazby jsou naopak bezvýznamné. Posléze tedy bude možné ze získaných výsledků vytvořit typologii pacientů a vyvinout pro ně účinnější léčbu. Veškerá pacientská data jsou uložena v databázovém systému MySQL. Návrh a tvorba databáze jsou také součástí této práce.

Klíčová slova

Metabolická data, databáze MySQL, MATLAB, neuronové sítě, samoorganizující se mapy, cluster

Abstract

This bachelor thesis deals with software development in MATLAB programming environment, which is able to classify a large dataset of complex metabolic pacient's data. The dataset was obtained in clinical examination of the pacients in Hradec Kralove hospital in years 2007-2010. Programme is based on the methods of artificial inteligence and its core is formed from the self-organizing map neural network architecture. Application is designed to process multi-dimensional input data and classify them into classes with similar parameters, so-called clusters. From graphical and numerical programme's outputs are doctors able to determine, which metabolic parameters are highly dependent and which parameters have no mutual relations. Afterwards it is possible to create typology of the pacients and evolve more effective therapy. The whole pacient's dataset is stored in MySQL database system. Design and formation of the database is also part of this bachelor thesis.

Key words

Metabolic data, MySQL database, MATLAB, neural networks, self-organizing maps, cluster

Seznam použitých symbolů a zkratek

BMI	body mass index	[kg/ m^2]
BSA	povrch lidského těla	[m^2]
EE	energetický obrat	[kcal/24hod]
GUI	grafické uživatelské rozhraní	
ID	identifikační číslo	
MySQL	databázový systém	
NCTOOL	neural network clustering tool	
RQ	respirační kvocient	[-]
SOM	self-organizing map	
UNS	umělé neuronové sítě	
VO2	spotřeba kyslíku	[ml/min]
VCO2	produkce oxidu uhličitého	[ml/min]

Obsah

1. Kapitola

Úvod

Zpracování dat získaných měřením je nezbytnou součástí nejen v průmyslové technické praxi, ale také v medicíně. Sledováním závislostí mezi metabolickými parametry a jejich správnou interpretací lze dosáhnout efektivnější léčby pacientů a zkvalitnit tak možnosti současného zdravotnictví. Vytvoření typologie pacientů s podobnými příznaky a metabolickými parametry by pomohlo vyvinout medicínské metody určené na míru různým typům pacientů. Metabolických procesů v lidském těle je nespočetné množství a jedná se o velmi rozsáhlý komplex jevů, které se navzájem ovlivňují. Otázkou je, jak tyto jevy klasifikovat. Jednoduchou klasifikaci pacientů na základě dvou parametrů je možné provést běžnými statistickými metodami. Problém ovšem nastává v momentě, kdy je zapotřebí pacienty klasifikovat na základě mnoha v principu odlišných metabolických parametrů nabývajících různých hodnot. Řešení se nabízí v podobě metod umělé inteligence, které jsou v současné době moderní dynamicky se rozvíjející oblastí počítačového inženýrství.

Tato bakalářská práce je věnována návrhu a realizaci programu vytvořeného v prostředí MATLAB, který je schopen při zachování určitých podmínek efektivně klasifikovat pacienty na základě mnoha vstupních metabolických parametrů. Jedná se o obecný algoritmus, jehož jádro se opírá o možnosti rozšíření MATLABu nazvané Neural Network Toolbox, a který je možné aplikovat na rozmanité druhy vstupních dat, primárně je však určen pro klasifikaci komplexních metabolických pacientských dat. Korektní odbornou interpretaci výstupů z programu ovšem může provést pouze specialista v daném oboru, u pacientských dat lékař.

Kapitola druhá pojednává o průběhu klinického vyšetření, na základě kterého byla získána veškerá data, která jsou analyzována, zpracovávána a klasifikována v této bakalářské práci. Součástí kapitoly je popis měřicího přístroje Deltatrac II, vstupních a výstupních dat a jeho kalibrace.

Třetí kapitola se zabývá možnostmi návrhu rozsáhlé databáze, ve které jsou veškerá metabolická pacientská data uložena. Popisuje výhody zvoleného databázového systému MySQL, způsob zprovoznění komunikace mezi tímto systémem a programovacím prostředím MATLAB, ve kterém jsou pacientská data zpracovávána, a možnosti komunikace na rozhraní MySQL – MATLAB.

Obsahem čtvrté kapitoly je realizace databáze. Vzhledem k tomu, že datové výstupy z měřicího přístroje Deltatrac II jsou v textovém formátu, je nutné nejprve vytvořit algoritmus, který z textových souborů vybere požadovaná data a upraví je před zápisem do databáze tak, aby byla kompatibilní s jazykem MySQL. Kapitola popisuje algoritmy pro tvorbu databáze a v závěru její výslednou strukturu.

V páté kapitole je provedena základní statistická analýza pacientských dat. Tato analýza je nutná z důvodu návrhu a realizace klasifikátoru, aby jej bylo možné optimalizovat pro vstupní pacientská data. Dále jsou v této kapitole uvedeny významy statistických nástrojů, které se uplatní i při testování klasifikačního software.

Šestá kapitola je věnována moderním algoritmům, které jsou používány ke klasifikaci dat. Vysvětluje významy pojmů neuronová síť a data clustering.

Stěžejní kapitolou této bakalářské práce je kapitola sedmá. Týká se návrhu a realizace klasifikačního programu, který je vytvořen v prostředí MATLAB v rámci jeho rozšíření Neural Network Toolbox. První část kapitoly popisuje manuální návrh neuronové sítě pomocí rozhraní Clustering Tool v MATLABu, možnosti vstupních dat a vysvětlení významu výstupů. Druhá část je věnována popisu algoritmu použitého v programu classifier.m, který je vytvořen speciálně pro potřeby této bakalářské práce.

Obsahem poslední kapitoly je testování vytvořeného software a interpretace grafických a numerických výsledků.

V závěru této práce je zhodnocen průběh její realizace, dosažené výsledky, jsou zdůvodněny výhody a nevýhody klasifikačního software a nastíněny další možnosti výzkumu v této oblasti zpracování dat.

2. Kapitola

Měření metabolických dat

Vstupní data, která jsou použita v této bakalářce práci, byla získána na základě měření metabolických parametrů pacientů nemocnice v Hradci Králové. Jedná se o poměrně jednoduché klinické vyšetření, kdy pacient podstoupí měření objemu spotřebovaného kyslíku a objemu vydechovaného oxidu uhličitého po dobu přibližně deseti minut. Z uvedených hodnot jsou poté přístrojem dopočítány hodnoty využití energetických substrátů. Na základě těchto údajů je možné klasifikovat a vytvořit určitou typologii pacientů v závislosti na stupni obezity, věku, reakci těla na trávení glukózy a mnoha dalších metabolických parametrech.

Obezita je závažným onemocněním především v důsledku vzniku komplikací, které následně mají řadu důsledků zdravotních, ale i ekonomických, sociálních a psychosociálních. [1] Při účinné léčbě vyvinuté na míru určitým typům pacientů lze stupeň obezity snížit či zcela potlačit. Z toho důvodu je účelné pacienty na základě metabolických dat typologicky klasifikovat.

2.1 Průběh měření

Pacient podstoupí výše uvedené klinické vyšetření. Ihned po skončení měření je mu podána glukóza. Po hodině odpočinku podstoupí pacient druhé měření a po další hodině měření třetí. Mezi těmito měřeními podstoupí pacient také krevní vyšetření. Výsledky tohoto krevního vyšetření však nejsou předmětem této bakalářské práce. Při zpracování a klasifikaci pacientských dat je sledován vliv podané glukózy na metabolické procesy. Mnoho pacientů však nepodstoupí tato měření třikrát během jednoho dne. Tato měření jsou ovšem z hlediska této práce bezvýznamná, přesto jsou součástí databáze pacientských dat, jak bude uvedeno v kapitole 4.

2.2 Popis měřicího přístroje

Metabolický monitor Deltatrac II je nepřímý kalorimetr, který byl vyvinut pro klinické použití, jakož i pro zdravotnický výzkum. Přístroj je schopen měřit metabolické parametry při umělé i spontánní plicní ventilaci. Rozličné oblasti provozu umožňují monitorování pacientů od novorozenců až po dospělé pacienty s extrémní nadváhou. Aby měřicí systém plnil svou determinovanou funkci, je nutné jej ve stanovených časových intervalech kalibrovat v souladu s manuálem k přístroji. [2] Přístroj slouží k přesnému stanovení látkové výměny a energetické spotřeby. Je využíván i pro mnohostranné vědecké výzkumy. Sleduje oxidační pochody látkové přeměny tuků a cukrů. [3]

Obr. 2.1 Metabolický monitor Deltatrac II [3]

2.3 Datový výstup z přístroje

Průběh celého měření je zaznamenáván do souboru v textovém formátu. Ukázka datového výstupu z přístroje je součástí přílohy I. této bakalářské práce. Soubor obsahuje následující náležitosti.

2.3.1 Zadané údaje o pacientovi

Před začátkem měření je nutné do přístroje zadat údaje, které budou sloužit k identifikaci pacienta v rámci klasifikace metabolických dat:

- Pohlaví
- Datum narození a věk
- Výška
- Hmotnost
- Vylučování dusíku
- Hodnota povrchu těla
- Basální metabolický poměr
- Velikost pasu – není obsaženo ve všech výstupech z měření
- Velikost boků – není obsaženo ve všech výstupech z měření
- Rodné číslo – není obsaženo ve všech výstupech z měření

2.3.2 Režim měření

Přístroj je schopen měřit metabolické parametry ve dvou módech [2]:

- Umělá plicní ventilace (RESPIRATORY MODE)
- Spontánní plicní ventilace (CANOPY MODE)

Všechna dostupná pacientská data byla získána na základě měření spontánní plicní ventilace. Podle velikosti pacienta je možné volit ze čtyř režimů měření, která se liší v hodnotě průtoku vzduchu:

Tabulka 2.1 Režimy měření při spontánní plicní ventilaci [2]

Váha	Rozsah měření	Průtok
< 3 kg	kojenec "baby"	3 l/min
3 - 20 kg	dítě "child"	12 l/min
20 - 120 kg	dospělý "adult"	40 l/min
> 120 kg	nadváha "obese"	80 l/min

2.3.3 Údaje o čase měření a kalibrace

Ve výsledném datovém výstupu z přístroje jsou zaznamenány časy, kdy měření začalo, skončilo, délka celého měření a doba přerušení. Dále je také uvedena doba trvání pohybových artefaktů. Součástí výstupu jsou i údaje o kalibraci přístroje, tedy čas a procentuální poměry kalibračního plynu.

2.3.4 Měřené a dopočítané veličiny

Hodnoty následujících parametrů jsou již získávány v průběhu měření. Přístroj tyto parametry zaznamenává každou minutu, ve výsledném datovém výstupu je však uvedena pouze střední hodnota, směrodatná odchylka vyjádřená také procentuelně. [2]

- Produkce oxidu uhličitého (V_{CO2}) v mililitrech za minutu (měřená proměnná).
- Spotřeba kyslíku (V_{O2}) v mililitrech za minutu (měřená proměnná).
- Respirační Kvocient (RQ) je vypočtená proměnná, která odpovídá podílu oxidu uhličitého ke spotřebě kyslíku.
- Energetický obrat (EE), který se udává v kilokaloriích za 24 hodin (vypočtená proměnná).
- Hodnoty využití energetických substrátů (karbohydráty, tuky a proteiny).

Data pro účely této práce byla získána v letech 2007-2010. V průběhu těchto let byl datový výstup postupně doplňován o další údaje. Při návrhu databáze pacientských dat (kapitola 4) je nutné vzít v potaz tuto skutečnost a data, která v prvních datových výstupech chybí, dopočítat či jinak doplnit.

2.4 Kalibrace přístroje

Výrobce doporučuje kalibrovat metabolický monitor Deltatrac II jednou denně (myšleno kalibrace plynů), k dosažení optimální přesnosti však nejlépe před každým měřením. Před kalibrací je důležité, aby se přístroj zahřál, čekací doba činí 30 minut. Je nutné vzít v potaz, že přístroj lze kalibrovat pouze v režimech vypnutého či přerušeného měření. [2]

2.4.1 Kalibrace plynů

Při kalibraci je nejprve automaticky prověřována základní úroveň pro senzor kyslíku a senzor kysličníku uhličitého. Přístroj je kalibrován plynem DatexEngstrom, který obsahuje 95% kyslíku a 5% CO_2. V případě použití jiného kalibračního plynu je nutné do přístroje zadat poměr těchto dvou plynů, který je uveden na kalibrační lahvi.

2.4.2 Kalibrace tlaku

Přístroj by podle výrobce měl být tlakově kalibrován každých 6 měsíců. Po provedené kalibraci tlaku by měla být vždy provedena navíc i kalibrace plynů.

2.4.3 Kalibrace průtoku

Metabolický monitor Deltatrac II je vybaven velmi stabilním generátorem průtoků, přesto je však nutné přístroj kalibrovat na průtok jednou za 2-3 měsíce. Je důležité, aby byla tato kalibrace provedena velmi pečlivě v souladu s manuálem dodávaným výrobcem.

2.5 Měření pohybových artefaktů

Přístroj Deltatrac II je schopen indikovat a následně zpracovávat pohybové a svalové artefakty. Tyto artefakty vznikají v případě, že pacient vykoná během měření prudší pohyb. Přístroj takto naměřená data vyjme z celkového vzorku dat, aby nemohla ovlivnit výsledek měření. Do datového výstupu zaznamená, jak dlouho byly pohybové a svalové artefakty měřeny.

2.6 Body Mass Index (BMI)

Pro účely statistického zpracování a klasifikace dat je nutné získat ještě jeden parametr, který přístroj nezaznamenává ani nedopočítává. Jedná se o Index tělesné hmotnost (v angličtině Body Mass Index, zkratka BMI), který vyjadřuje stupeň obezity pacienta. Tento parametr je snadné dopočítat z pacientovy výšky a hmotnosti ze vztahu [4]:

$$BMI = \frac{hmotnost\ (kg)}{výška\ (m^2)}$$

Z tabulky BMI je pak možné vyčíst, zda má pacient podváhu, zda je jeho hmotnost vzhledem k výšce normální nebo zda je obézní, popřípadě jakým stupněm obezity trpí. V následující tabulce jsou uvedeny rozsahy hodnot BMI, podle kterých se určuje stupeň obezity.

Tabulka 2.2 Kategorie BMI [4]

Pořadí kategorie	Kategorie	Rozsah BMI – kg/m^2
0	těžká podvýživa	BMI ≤ 16,5
1	podváha	16,5 – 18,5
2	ideální váha	18,5 – 25
3	nadváha	25 – 30
4	mírná obezita	30 – 35
5	střední obezita	35 – 40
6	morbidní obezita	BMI > 40

Označení pořadí kategorie začíná nulou z toho důvodu, že žádný z pacientů v databázi není klasifikován jako těžce podvyživený, ve statistické analýze (podkapitola 5.3) tedy číslování skupin začíná až od kategorie 1 – podváha, viz obr. 5.3.

BMI je metoda spíše jen orientační. Přesnější je stanovení složení těla. Je obecně známo, že jsou pacienti s vyšším BMI a relativně nižším % tuku. Jsou to spíše atletické typy. U mužů je normální zastoupení tuků 10 – 25%, u žen 18 – 30%. [1]

Přesto je hodnota BMI při klasifikaci pacientských dat získaných pro účely této bakalářské práce významným parametrem vzhledem k tomu, že drtivá většina vyšetřených pacientů je v pokročilém stupni obezity, více v podkapitole 5.2. V závislosti na míře obezity lze sledovat rychlost metabolických procesů po podání glukózy (podkapitola 2.1).

3. Kapitola

Databázové prostředky pro medicínská data

Metabolická pacientská data, která jsou zpracovávána, statisticky vyhodnocována a experimentálně klasifikována v této bakalářské práci, jsou uložena v databázovém systému MySQL. Jedná se o multiplatformní databázový systém, jehož licence k užívání je bezplatná. Mezi jeho hlavní výhody patří vysoký výkon a spolehlivost. [5] Systém MySQL byl pro účely této práce zvolen proto, že používané příkazy pro komunikaci s databází jsou jednoduché a intuitivní. Hlavním důvodem volby právě tohoto databázového systému je však především skutečnost, že MySQL je kompatibilní s prostředím MATLAB, ve kterém jsou data zpracovávána a klasifikována a vyhodnocována.

3.1 Rozhraní MySQL – MATLAB

Před zapisováním dat do databáze je nutné zprovoznit konektor mezi databází a MATLABem. Jedná se o m-file skript, ve kterém jsou specifikovány následující parametry [6]:

- název serveru, ke kterému se MATLAB v rámci systému MySQL připojuje
- název databáze
- název uživatele
- heslo do databázového systému
- úplná cesta k pomocnému JAVA konektoru
- druh driveru

Klíčovým příkazem pro připojení k MySQL databázi je příkaz „database“, v jehož argumentu jsou všechny výše uvedené parametry. Připojení k databázi je možné ověřit příkazem isconnection(název_konektoru) zadaným do příkazového řádku prostředí MATLAB. Připojení k databázi vytvořené v rámci této bakalářské práce je možné realizovat pomocí naprogramovaného skriptu bc_1_database_connection.m.

3.2 Možnosti komunikace s databází

Jakmile je MATLAB připojen k MySQL serveru, nabízí se dvě možnosti komunikace:

- Zápis dat do databáze – příkaz exec(požadavek, konektor) v prostředí MATLAB.
- Získávání dat z databáze – příkaz fetch(požadavek, konektor).

Postup při komunikaci s databází je takový, že nejprve je v proměnné uveden požadavek, který má databáze vykonat a následně použit jeden z výše uvedených příkazů. Datový typ této proměnné je řetězec. Syntaxe požadavku na databázi musí být v souladu s jazykem MySQL,v opačném případě nebude příkaz vykonán.

Před zápisem dat do databáze je nejprve nutné vytvořit tabulku, ve které je specifikován počet sloupců a dále jaké datové typy je možné v jednotlivých sloupcích ukládat.

V případě zápisu do databáze je nutné vstupní data upravit tak, aby odpovídala datovým typům sloupců v tabulce. Pokud jsou některá data zapisovaná do databáze v nesprávném formátu, celý řádek, ve kterém jsou tato nesprávná data obsažena, je ze zápisu vynechán. Chybu v zápisu je možné ověřit v atributech příkazů fetch a exec, konkrétně atribut „message“. [6] Jakmile je prázdný, došlo ke správnému vykonání požadovaného příkazu.

3.3 Export databáze

Celou databázi pacientských dat je z důvodu vytvoření zálohy nutné exportovat z databázového systému. V operačním systému Windows je možné databázi exportovat a zálohovat ze serveru MySQL pomocí příkazu mysqldump v systémové konzoli. [7]

3.4 Další možnosti databázových prostředků

Dalšími variantami, jak ukládat metabolická pacientská data, jsou databázové systémy Microsoft SQL server a Access, Oracle, Firebird a mnoho dalších. Pro účely této práce byl ovšem zvolen databázový systém MySQL kvůli jeho snadné a intuitivní komunikaci s prostředím MATLAB.

4. Kapitola

Návrh databáze pro klasifikaci metabolických dat

Pacientská data, jejichž zpracování a následná klasifikace jsou cílem této bakalářské práce, byla získána měřením na přístroji Deltatrac II v letech 2007, 2008, 2009 a 2010. V těchto letech bylo na oddělení prof. Martiníka v nemocnici v Hradci Králové provedeno 10929 měření. Soubor dat ke zpracování je tedy mimořádně obsáhlý.

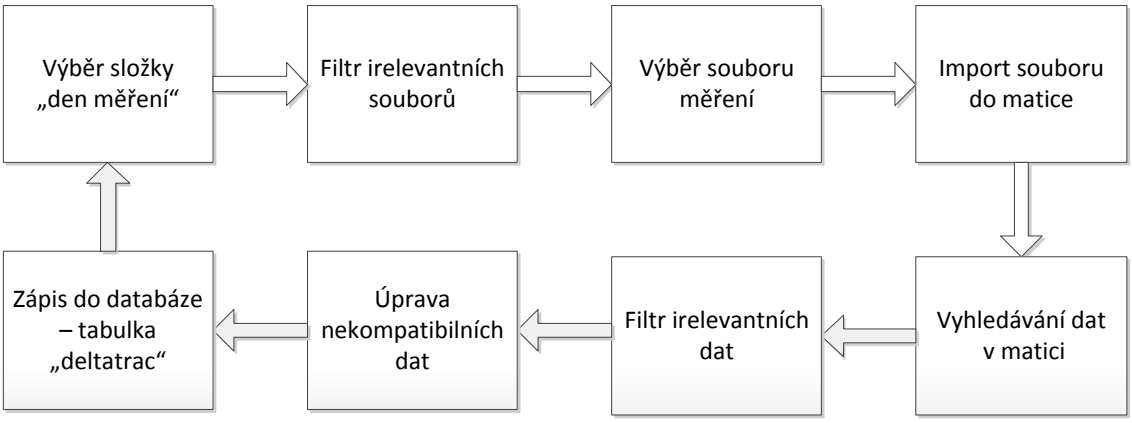
Jak je uvedeno v podkapitole 2.3, výstupní soubor z přístroje Deltatrac II je v textovém formátu. Tento formát je z hlediska zpracování dat nevhodný, protože naměřené metabolické parametry nejsou uloženy v buňkách, ale jsou součástí textu. Proto je nutné vytvořit algoritmus, který je schopen z textového souboru vybrat požadované informace a následně je zapsat do databáze.

4.1 Druhy vstupních dat

Při návrhu databáze je nutné vzít v potaz, že se v průběhu let se výstupní soubor z přístroje Deltatrac II měnil. Soubory z let 2007 a 2008 neobsahují rodná čísla pacientů, která jsou jednoznačným parametrem pro určení pacientů. Část souborů obsahuje informace nepotřebné v rámci této práce. Některé výstupy z měření jsou neúplné či poškozené a je tedy nutné je odstranit, aby nemohly ovlivnit výslednou klasifikaci metabolických dat.

4.2 Import pacientských dat do databáze

Celý proces importu dat probíhá v prostředí MATLAB. Pro potřeby importu dat do databáze je naprogramován skript bc_2_main_parse.m, jehož struktura je zobrazena na obr 4.1.



Obr. 4.1 Algoritmus importu dat do databáze - tabulka „deltatrac“

Na začátku řetězce je vybrána složka měření. Tato složka obsahuje data, která byla naměřena v tentýž den. Dále jsou odfiltrovány soubory, které neodpovídají požadovaným parametrům a jsou neúplné či poškozené. Po výběru souboru dojde k jeho importu do matice. V cyklu, ve

kterém jsou porovnávány řetězce, dochází k průchodu celou maticí a vytvoření proměnných, do kterých jsou uloženy veškeré požadované informace. Před zápisem do databáze je nutné převést některá data tak, aby byla kompatibilní s jazykem databázového systému MySQL. Jedná se především o formáty časů měření, dat narození a rodných čísel. U souborů, které neobsahují rodné číslo, je toto číslo vytvořeno z data narození, přičemž koncovka je doplněna nulami. U žen je k rodnému číslu přičtena hodnota 5000 tak, aby bylo dosaženo standardního tvaru. Jakmile jsou veškerá data připravena, jsou zapsána do databáze, konkrétně do tabulky „deltatrac“. Celý cyklus je opakován, dokud nedojde k průchodu všemi dostupnými vstupními soubory.

Každému měření je přiřazeno ID, které je zároveň primárním klíčem databáze. Na základě tohoto primárního klíče lze při návrzích dalších tabulek v databázi vybírat konkrétní měření a získávat z nich požadované informace. ID v tabulce „deltatrac“ tedy jednoznačně určuje jedno měření jednoho pacienta v jednom dni.

Tabulka „deltatrac“ má 10213 řádků, přičemž jeden řádek obsahuje data z jednoho měření pacienta. Srovnáme-li tento počet s počtem vstupních souborů (10929), je zřejmé, že přibližně 700 souborů z měření neprošlo filtrem z důvodu poškození či neúplnosti.

4.3 Selekce požadovaných pacientských dat

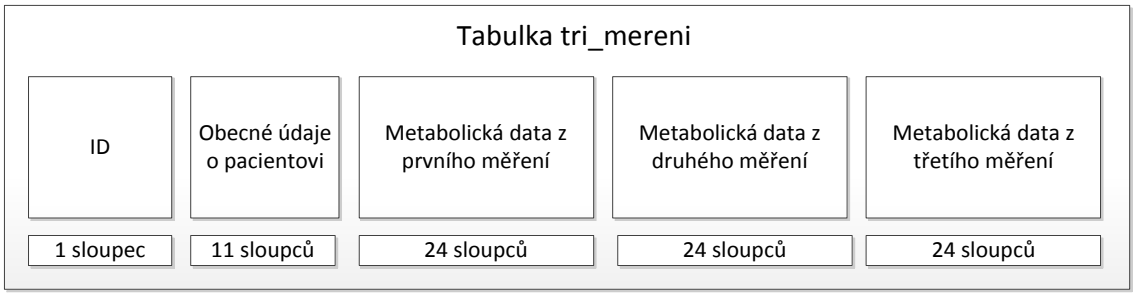
Pro účely experimentu klasifikace metabolických dat je nutné vybrat pouze takové pacienty, kteří podstoupili měření přesně 3x v daném dni, aby bylo možné sledovat vliv podání glukózy na metabolické procesy, viz kapitola 2.

K nalezení těchto pacientů slouží skript bc_3_measurements_per_day.m napsaný v prostředí MATLAB, který prochází tabulku všech dostupných měření „deltatrac“ a určuje, kolikrát pacienti v daný den měření podstoupili. Počty měření v jednom dni jsou zaznamenávány v tabulce „pocet_mereni“, a zároveň jsou do ní ukládána původní ID jednotlivých měření pacientů z tabulky „deltatrac“ tak, aby bylo možné jednoznačně určit, o která měření se jedná. V následující tabulce je uveden přehled počtů měření.

Tabulka 4.1 Počet měření v jednom dni

Počet měření v jednom dni	Počet položek
1	3380
2	1372
3	1173
4	15
5	2

V závěrečné fázi návrhu databáze pro klasifikaci metabolických pacientských dat jsou vybráni pacienti, kteří měření podstoupili 3x v jednom dni. Tabulka „tri_mereni“ je koncipovaná tak, že v prvních sloupcích jsou zaznamenány obecné údaje o pacientovi, například rodné číslo, hmotnost, věk, BMI apod. Následují sloupce obsahující hodnoty z jednotlivých měření v daném dni. Z této tabulky vychází veškerá statistická zpracování a návrh experimentu klasifikace metabolických dat, který bude popsán v následujících kapitolách.



Obr. 4.2 Struktura tabulky „tri_mereni“

Tabulka „tri_mereni“ je vytvořena na základě skriptu bc_4_three_measurements_per_day.m, který je opět naprogramován v prostředí MATLAB.

4.4 Struktura výsledné databáze

Celá databáze v systému MySQL tedy obsahuje tři tabulky, přičemž stěžejní je tabulka „tri_mereni“. Dále byly, kromě již známých parametrů a metabolických hodnot z výstupu přístroje Deltatrac II, dopočítány hodnoty Body Mass Indexu (BMI), na základě kterého je lze klasifikovat pacienty. Pro tyto účely je v rámci této práce naprogramována funkce body_mass_index.m. Dále byl do této tabulky dopočítán bazální metabolický index, který znázorňuje pacientovu schopnost spalování kalorií v průběhu jednoho dne. Funkce basal_metabolic_rate.m vytvořená pro potřeby importu dat do databáze je součástí elektronických příloh této bakalářské práce. Na základě hodnoty bazálního metabolického indexu mohou lékaři určit pokročilé souvislosti mezi metabolickými parametry.

Tabulka 4.2 Přehled tabulek v MySQL databázi – název „bakalarka“

Název tabulky	Primární klíč	Počet řádků	Počet sloupců	Poznámka
deltatrac	ID	10213	42	Všechna dostupná měření pacientů.
pocet_mereni	ID	6442	9	Počty měření pacientů v jednom dni.
tri_mereni	ID	1173	84	Pacienti, kteří byli změřeni 3x v jeden den.

5. Kapitola

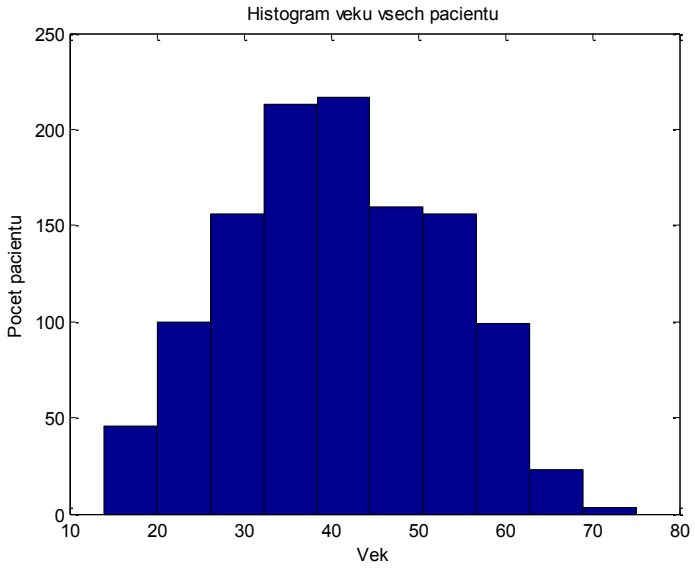
Statistické zpracování metabolických parametrů

Pro návrh experimentu klasifikace metabolických dat je důležité vědět, jakých hodnot nabývají jednotlivé metabolické parametry, aby bylo možné sledovat jejich vzájemné souvislosti, a tím zlepšit interpretaci výsledků získaných v průběhu procesu klasifikace. Veškeré statistické zpracování dat je řešeno v prostředí MATLAB. Pro účely této práce ovšem postačí pár elementárních nástrojů statistiky. Analyzována jsou metabolická data pacientů, kteří podstoupili měření 3x v jednom dni. Tato data jsou uložena v tabulce „tri_mereni“, viz podkapitola 4.3. Skript v MATLABu nejprve načítá vybrané metabolické parametry z databáze, a poté je srovnává a zobrazuje do přehledné grafické podoby.

Jedním z nejdůležitějších údajů o pacientech, na kterém staví následný experiment klasifikace metabolických dat, je stupeň jejich obezity, který je udáván hodnotou BMI (podkapitola 2.6). Intervalové rozdělení četnosti se často znázorňuje graficky pomocí histogramu nebo polygonu četnosti. Při kreslení histogramu vynášíme na osou x intervaly a na osu y četnosti v těchto intervalech. Obdélníčky se stranami odpovídajícími intervalu hodnot a dosažené četnosti vytvoří histogram. [8]

5.1 Věkové rozložení pacientů

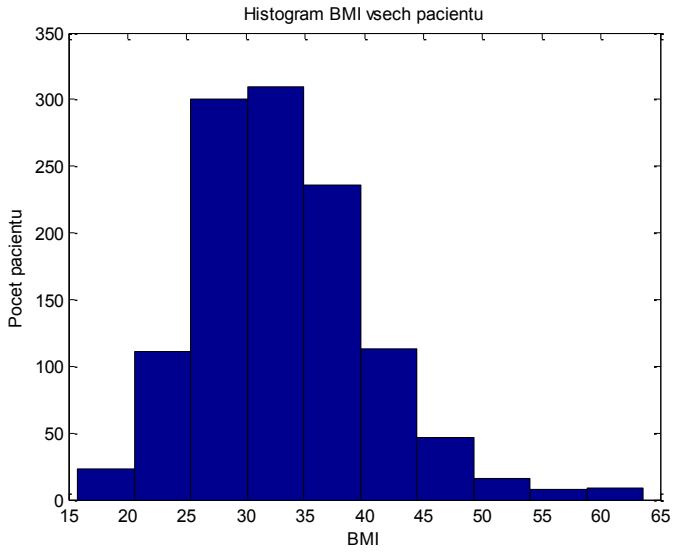
První histogram zobrazuje věkové spektrum pacientů, kteří měření podstoupili. Z grafu je patrné, že rozložení věku pacientů je vesměs rovnoměrné a má charakter Gaussova rozložení. Největší skupinu tvoří pacienti ve středních letech, naopak pacienti nad 70 let jsou pouze dva.



Obr. 5.1 Intervalové četnosti věku pacientů.

5.2 Body Mass Index pacientů

Pomocí následujícího histogramu je vzorek měření pacientů rozdělen na rovnoměrné skupiny podle hodnoty Body Mass Indexu (BMI). Data jsou takto rozdělena z toho důvodu, že stupeň obezity, který je tímto Indexem udáván, má vliv na rychlost metabolických procesů a na kvalitu využití energetických substrátů. Následující histogram zobrazuje intervalové četnosti BMI.



Obr. 5.2 Intervalové četnosti Body Mass Indexu pacientů.

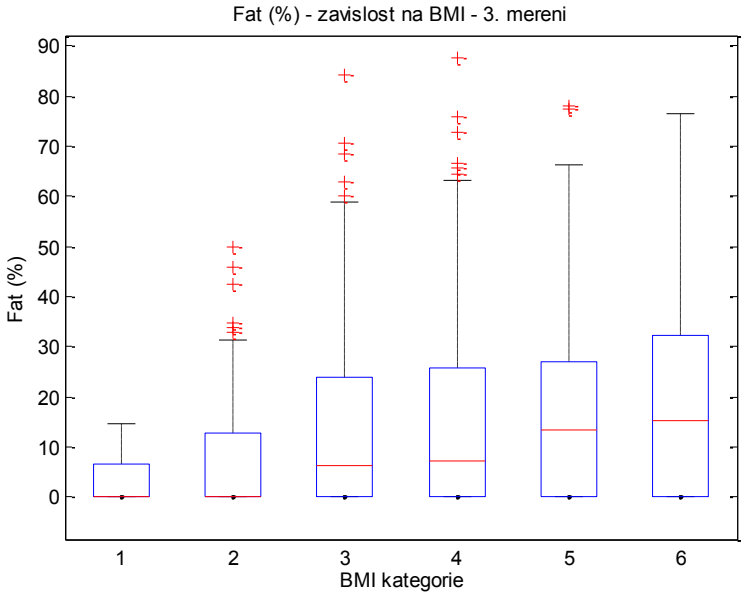
Z histogramu je patrné, že normální hodnotu BMI, respektive mírnou podváhu, má pouze přibližně 10% pacientů, kteří podstoupili měření energetického výdeje. Naopak nejvyššího stupně obezity (BMI > 40) dosahuje přibližně 15% pacientů. Je tedy patrné, že drtivá většina pacientů trpí určitým stupněm nadváhy.

5.3 Statistická analýza využití energetických substrátů

Dalším statistickým nástrojem, který je použit v této práci, je tzv. krabicový graf. Jedná se o přehledné zobrazení základních statistických parametrů tak, aby bylo vzájemné srovnání veličin co nejjednodušší.

Červená čára určuje medián ze vzorku dat, tedy hodnotu, která dělí data seřazená podle velikosti na dvě stejně početné části. Horní a dolní hrana „krabice“ označuje horní, respektive dolní kvartil. Mezi horním a dolním kvartilem je tzv. interkvartilový interval. Konce úseček jsou ve vzdálenosti 1,5 násobku rozpětí interkvartilového intervalu. [9] Křížky, které jsou vzdálené dál než hranice úseček, představují naměřená data, která mohla být získána na základě nepřesného či špatně provedeného měření, tzv. odlehlá měření. Mohou však také vyjadřovat anomálie v metabolických pacientských datech a je tedy nutné je brát při klasifikaci v potaz.

Na následujícím krabicovém grafu je zobrazeno využití tuků z energetických substrátů ve třetím měření, tedy dvě hodiny po podání glukózy. Grafy jsou rozděleny podle hodnot BMI v obecně definovaných skupinách, viz tabulka 2.2 v podkapitole 2.6.



Obr. 5.3 Závislost využití tuků v energetických substrátech na BMI – 3. měření pacientů

Z grafů je patrné, že s rostoucí obezitou se rozšiřuje i interkvartilový interval a medián hodnot se procentuelně zvyšuje. Nejvíce odlehlých měření obsahují nejvíce zastoupené skupiny pacientů. V případě, že by byla vykreslena a v jednom grafu srovnána data využití tuků z energetických substrátů z jednotlivých měření, bylo by možné sledovat vliv podané glukózy na rychlost metabolických procesů.

5.4 Další statistické zpracování pacientských dat

Podobně jsou pomocí vytvořeného skriptu bc_5_statistic.m v MATLABu pro medicínské účely vykresleny a srovnány i ostatní důležité metabolické parametry. Uvedené metabolické parametry jsou ovšem děleny pouze na základě stupně obezity (BMI). Databáze pacientských dat však umožňuje sledovat vzájemné souvislosti všech metabolických parametrů a představuje tedy množství oblastí, které lze zkoumat. Na základě důkladné analýzy všech možných vztahů mezi metabolickými jevy je možné přesněji diagnostikovat pacienty a zlepšit tak kvalitu zdravotnictví. Analýzu a následnou interpretaci metabolických vztahů však mohou provést pouze lékaři.

6. Kapitola

Algoritmy klasifikace dat

Pohledů na problematiku klasifikace dat je celá řada. Jednoduché klasifikační problémy lze řešit běžnými statistickými metodami. Problémem je ovšem klasifikace dat metabolických, kdy je nutné klasifikovat vstupní data na základě mnoha navzájem odlišných parametrů. Tato práce se zabývá klasifikací pomocí moderních metod umělé inteligence, konkrétně samoorganizujícími se umělými neuronovými sítěmi. V textu této bakalářské práce se pojmem neuronové sítě rozumí umělé neuronové sítě.

6.1 Umělé neuronové sítě (UNS)

Umělé neuronové sítě jsou struktury, které jsou inspirovány svými biologickými vzory. Jejich hlavním úkolem je simulovat a implementovat některé funkce lidského mozku, především schopnost učení a adaptace. [11] Jsou to matematické modely neuronových sítí živých organismů. Simulace neuronových sítí vykazují překvapivě velmi slušné výsledky. Problémem při jejich nasazování je ovšem omezení neuronových sítí, ať již z výpočetních tak i z hardwarových důvodů. [12] Podle toho, jak jsou neurony, což jsou základní prvky těchto sítí, topologicky a funkčně uspořádány, rozlišujeme tzv. architektury UNS. Architektura použitá pro potřeby této práce se nazývá samoorganizující se mapa, anglicky self-organizing map.

6.2 Self-organizing map

Samoorganizující se mapy (SOM) jsou neuronové sítě, které se používají především pro klasifikaci dat (i velkých souborů), pro jejich kompresi (snižování počtu dat a dimenze dat) a v neposlední řadě pro možnost vizualizace dat, včetně vizualizace základních vztahů. [13] Tento typ architektury neuronových sítí je často označován také pojmem Kohonenovy sítě nebo Kohonenovy samoorganizující se mapy. Architektura je pojmenována po svém autorovi, jímž je finský profesor Teuvo Kohonen z Helsinské technologické univerzity.

Základní myšlenka SOM vychází z poznatku, že i lidský mozek používá pro uchovávání a zpracování dílčích informací vnitřní reprezentaci dat, která má nižší dimenzi, než je původní dimenze dat. Podstata tohoto typu neuronové sítě spočívá v tom, že se vstupní referenční vektory sdružují do skupin podle svých navzájem podobných vlastností a zobrazují se jako shluky (anglicky „clustery“) v tzv. elastické vrstvě neuronové sítě. Pojem elastická znamená, že v ní dochází ke změně vah (synapsí) mezi jednotlivými neurony. Během procesu trénování dochází ke kompresi informací při zachování nejdůležitějších topologických vztahů a vzdáleností. [13]

Self-organizing map je velmi specifická architektura z hlediska své topologie. Zatímco u jiných architektur jsou neurony spojeny každý s každým mezi jednotlivými vrstvami, u SOM jsou

jednotlivé neurony spojeny pouze se sousedními neurony. [12] Další důležitou vlastností SOM je skutečnost, že neuron nemá přenosovou funkci, která u běžných neuronů v jiných architekturách definuje proces generování výstupu z neuronu, ale pouze se přepočítávají vzdálenosti vstupního vzoru ke vzoru zakódovaného ve vahách daného neuronu. V postupu předkládání vektoru vstupních dat se mění excitace výstupních neuronů tak, že u jednoho neuronu jeho výstup roste a u ostatních klesá. [11]

Charakteristický rysem samoorganizujících se map je schopnost uspořádat množinu přicházejících vstupů do struktury s předem danou topologií a dimenzemi. Toto uspořádání probíhá samovolně, bez regulace pomocí tzv. učitele. [13] Proto se při trénování modelu neuronové sítě s architekturou SOM mluví o tzv. učení bez učitele, který by neuronové síti signalizoval, jak mají vypadat výstupy.

6.3 Trénování self-organizing map

Učení je založeno na schopnosti rozeznat ve vstupních vektorech stejné nebo blízké vlastnosti a třídit přicházející vektory podle nich. Podobné vektory sdružuje do shluků (tzv. clusterů) v mapě. [13]

Obecně je možné učení SOM realizovat prostřednictvím následujících kroků [12]:

- **Krok 1. Inicializace**
  Nastavíme váhy w_{ij} na malé náhodné počáteční hodnoty.
- **Krok 2. Předložení vzoru**
  Předložíme nový trénovací vzor $X(t) = \{x_0(t),\ x_1(t), \dots ,\ x_{N-1}(t)\}$ na vstup neuronové sítě.
- **Krok 3. Výpočet vzdálenosti vzorů**
  Jak je uvedeno v podkapitole 6.2, učení SOM probíhá přepočtem vzdáleností. Výpočet vzdálenosti nebo také podobnosti d_j mezi předloženým vzorem a všemi výstupními neurony j je dle vztahu:
  $$d_j = \sum_{i=0}^{N-1} \left[x_i(t) - w_{ij}(t)\right]^2$$
  kde $x_i(t)$ jsou jednotlivé elementy vstupního vzoru $X(t)$ a $w_{ij}(t)$ jsou váhy mezi i-tým vstupem a j-tým neuronem, které představují zakódované vzory.
- **Krok 4. Výběr nejbližšího neuronu (souseda)**
  Vybereme výstupní neuron $j*$, který splňuje následující podmínku a odpovídá tak nejpodobnějšímu neuronu:
  $$d_{j*} = \min(d_j)$$
- **Krok 5. Přizpůsobení vah**
  Přizpůsobíme váhy pro daný neuron $j*$ a jeho okolí $N_{j*}(t)$, tj. pro všechny neurony ležící uvnitř tohoto okolí podle následujícího vztahu:
  $$w_{ij}(t+1) = w_{ij}(t) + \eta(t)\left[x_i(t) - w_{ij}(t)\right]$$

Na začátku se hodnota váhy volí blízko jedné a postupem času se zmenšuje k nule. Nesmíme přitom při učení zapomenout na postupné snižování okolí až na předem definované minimální okolí.

- **Krok 6. Pokračování učícího procesu**
  Při procesu učení, pokud jsme nevyčerpali všechny vzory, kterými chceme síť naučit, nebo jsme neprošli požadovaný počet trénovacích kroků, resp. nedosáhli požadované přesnosti, přejdeme ke kroku 2. V opačném případě, kdy je už síť naučena na všechny trénovací vzory, můžeme skončit.

Uvedený trénovací postup je však popsán pouze obecně. Každé programovací prostředí, včetně Neural Network Toolboxu v MATLABu (kapitola 7), používá k trénování neuronové sítě svůj specifický algoritmus, který se může v jednotlivých krocích lišit.

Je důležité si uvědomit skutečnost, že počáteční inicializace vah mezi jednotlivými vstupy je vždy založena na náhodném přiřazování hodnot. To má za následek, že výsledky z neuronové sítě jsou při každém výpočtu vždy trochu odlišné. To je případ i klasifikačního software, jehož návrh a realizace je obsahem kapitoly 7. Clustery (shluky dat s podobnými vlastnostmi) obsahují po průběhu klasifikačního programu vždy trochu jiný počet pacientů.

V případě reálných dat (což dostupná pacientská data jsou) bývá vhodné použít normování. Zlepšuje totiž numerickou přesnost. [13] Tento proces je detailněji popsán v podkapitole 7.3.

7. Kapitola

Realizace SW pro klasifikaci dat

Neuronové sítě, jakožto oblast umělé inteligence, se řadí mezi moderní metody počítačového inženýrství, kterými lze řešit úlohy neřešitelné konvenčními počítačovými algoritmy. Jak je uvedeno v kapitole 6, klasifikační software je realizován v prostředí MATLAB. Rozšíření nazvané Neural Network Toolbox je velmi silným nástrojem pro řešení komplexních problémů v oblasti analýzy a zpracování dat. Kapitola obsahuje popis, jakými způsoby lze manuálně či automaticky využívat možnosti Neural Network Toolboxu.

7.1 Neuronové sítě v prostředí MATLAB

Pro práci s neuronovými sítěmi v prostředí MATLAB je nezbytné, aby disponoval rozšířením Neural Network Toolbox. V tomto Toolboxu (knihovna programových funkcí) jsou obsažena čtyři základní rozhraní, z nichž jedno je určeno pro tzv. Data Clustering, v češtině „Shlukování dat“. Klasifikátor je založen na možnostech právě tohoto rozhraní. Jádro rozhraní Clustering Tool tvoří poměrně specifická architektura modelu neuronové nazvaná Self-organizing map (SOM), která je detailně popsána v kapitole 6.

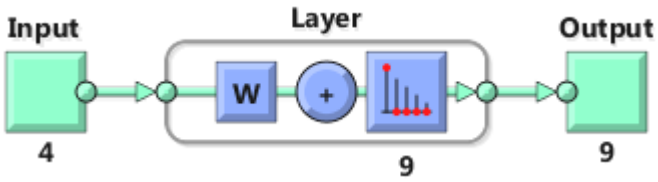
Je důležité si uvědomit, že při použití rozhraní Clustering Tool jsou klasifikované vzorky dat součástí klasických množin (clusterů), do kterých buď patří (logická 1) nebo nepatří (logická 0). Při klasifikaci tedy nedochází k určování míry příslušnosti k jednotlivým množinám, jako například u Fuzzy Clusteringu.

7.2 Self-organizing map v prostředí MATLAB

V prostředí MATLAB lze manuálně vyvolat GUI rozhraní Clustering Tool použitím příkazu „nctool“ (zkratka pro Neural Network Cluster Tool) přímo v příkazovém řádku. Základní nabídka obsahuje seznámení s možnostmi tohoto rozhraní. Přesunem na další nabídku (tlačítkem Next) lze vybrat vstupní data z workspace, je však nezbytné data normovat, viz následující část 7.3 této kapitoly. Je důležité dbát na orientaci matice vstupních dat tak, aby byl počet vzorků (samples) vyšší než počet tříd, do kterých jsou rozděleny (elements). Pro účely klasifikace pacientských dat jsou za vzorky považováni jednotliví pacienti a za třídy (elementy) jsou považovány parametry těchto pacientů, například věk, RQ, EE, BMI apod.

Jakmile jsou vzorky korektně importovány na vstup neuronové sítě, je nutné specifikovat topologii SOM. Výrazem topologie je v tomto smyslu myšlen počet neuronů, které tvoří neuronovou síť. Počet neuronů sítě definuje, do kolika shluků (clusterů) bude neuronová síť klasifikovat vstupní data. Obvykle se volí čtvercové matice počtu neuronů, například 3x3. Počet neuronů, a tedy také shluků, je v tomto případě 9.

Na následujícím diagramu je zobrazen model samoorganizující se mapy. Blok Input popisuje strukturu vstupních dat, která jsou rozdělena do čtyř tříd. Blok Layer popisuje strukturu SOM, která má v tomto případě rozměr 3x3, obsahuje tedy 9 neuronů. Poslední blok Output popisuje, do kolika shluků (clusterů) bude SOM klasifikovat vstupy.



Obr. 7.1 Diagram neuronové sítě Self-organizing map v nctool

V dalším kroku je možno přistoupit k tzv. trénování modelu neuronové sítě, které lze spustit v následující nabídce. Při trénování sítě jsou tzv. synapsím (spojům mezi jednotlivými neurony) přiřazeny tzv. váhy, které determinují, jak se bude shlukování vzorků na základě podobností provádět. Průběh trénování je zobrazen v nabídce Neural Network Training („nntraintool“), která se otevře ihned po spuštění tréninku. Nabídka trénování je zobrazena v příloze II. Po ukončení trénování (přednastaveno je 200 iterací) lze manuálně zobrazit výstupní grafy samoorganizující se mapy. V případě, že uživatel není spokojen s výsledky natrénované neuronové sítě, se nabízí několik možností:

- Opětovné spuštění trénování modelu neuronové sítě
- Změna topologie neuronové sítě (počet neuronů)
- Snížení či zvýšení počtu tříd, do kterých jsou rozdělena vstupní data
- Výběr většího vzorku vstupních dat (s rostoucím počtem vstupních dat je dosaženo kvalitnější natrénování neuronové sítě)

Jakmile jsou výsledky z neuronové sítě uspokojivé, lze provést dodatečné testy tím, že se na vstup neuronové sítě importují další vzorky (například pacienti), viz následující kapitola.

V poslední nabídce rozhraní Clustering Tool je možné uložit získané výsledky do workspace tak, aby s nimi šlo dále pracovat. Dále je možné vygenerovat jednoduchý m-file skript, kterým lze celý postup návrhu neuronové sítě zopakovat. Po ukončení práce lze GUI rozhraní Clustering Tool zavřít tlačítkem Finish.

7.3 Normování vstupních dat

Vzhledem k tomu, že parametry vstupních dat nabývají různých hodnot, je nezbytné veškerá data přivedená na vstup neuronové sítě normovat tak, aby měla stejný vliv na nastavení synapsí mezi neurony sítě. Tuto operaci lze provést pomocí funkce clust_normalize.m, která je součástí volně dostupného Fuzzy Clustering Toolboxu pro prostředí MATLAB. [10] Prvním argumentem této funkce jsou vstupní data, která je však nutné zadávat pouze po jednotlivých

třídách (například BMI pacientů). Druhým parametrem je volba typu normování. Klasifikační software vyžaduje, aby byla vstupní data normována lineárně v intervalu 0 až 1. Pokud se tedy například zmíněné BMI pacientů pohybuje v intervalu 17 až 63, hodnotě 17 je přiřazena normovaná hodnota 0, naopak hodnotě BMI 63 je přiřazena normovaná hodnota 1. Všechny ostatní hodnoty BMI jsou poté lineárně přepočítány na zmíněný interval 0 až 1. Tento typ normování lze u funkce clust_normalize.m nastavit použitím druhého parametru „range".

7.4 Grafické výstupy ze SOM

Pro nenumerickou analýzu klasifikovaných dat má rozhraní Clustering Tool přehledně zpracované grafické výstupy. Při manuálním použití tohoto rozhraní (příkazem nctool v příkazovém řádku) lze tyto grafy zobrazit použitím příslušných tlačítek. Základem každého zobrazení jsou hexagonální útvary představující jednotlivé neurony (clustery). Co se týče číslování neuronů, první neuron je v grafech umístěn vždy v levém dolním rohu. Naopak poslední neuron je umístěn v pravém horním rohu. Významy nejdůležitějších grafů jsou shrnuty v následujících podkapitolách.

7.4.1 Graf vzdáleností okolních vah

Toto zobrazení (v angličtině nazvané SOM Neighbor Weight Distances) přehledně znázorňuje, jak jsou jednotlivé clustery (shluky klasifikovaných dat) navzájem podobné. Mezi modrými hexagonálními útvary (neurony neboli clustery) je míra podobnosti rozlišena barevně. Čím světlejší je vazba mezi dvěma neurony, tím podobnější jsou clustery, mezi nimiž se tato vazba nachází. Naopak čím tmavší je vazba mezi clustery, tím se podobnost clusterů snižuje.

7.4.2 Graf počtu vzorků v clusterech

Zobrazení SOM Sample Hits znázorňuje, kolik vzorků vstupních dat obsahují jednotlivé clustery. Číslo uprostřed hexagonálních útvarů určuje počet vzorků v clusteru.

7.4.3 Graf vah jednotlivých vstupů

Zobrazení SOM Input Planes sestává z tolika grafů, kolik je tříd vstupních vzorků. Pokud tedy jsou například pacienti klasifikování podle věku, hodnoty BMI a hodnoty respiračního kvocientu, obsahuje zobrazení tři grafy. V těchto grafech je znázorněno, které parametry vstupních dat jsou navzájem závislé. Čím podobnější barvy v jednotlivých grafech jsou v daném neuronu, tím více jsou porovnávané parametry vstupních dat vzájemně závislé.

7.4.4 Graf pozic vah

Zobrazení SOM Weight Positions obsahuje tři základní prvky:

- Zelené body – reprezentují jednotlivé vzorky vstupních dat (například pacienty)
- Modré hexagonální útvary – reprezentují neurony (clustery)

- Červené vazby (úsečky) – zobrazují, které neurony jsou vedle sebe, jejich velikosti určují vzdálenosti mezi danými neurony

V tomto zobrazení se kolem neuronů shlukují klasifikovaná data. Síť neuronů spojená červenými vazbami je rozprostřena v prostoru klasifikovaných dat. Problémem však je, že červené vazby se mnohdy překrývají, protože je klasifikován vícerozměrný prostor. Tento problém je ovšem vyřešen ve funkci classifier.m, která je popsána v následující podkapitole.

Veškeré další údaje ke grafickým výstupům z neuronové sítě s architekturou self-organizing map jsou dostupné v nápovědě k programovacímu prostředí MATLAB.

Manuální návrh neuronové sítě bohužel neposkytuje možnost numerické analýzy získaných výsledků. Uživatel tedy neví, které konkrétní vzorky vstupních dat jsou klasifikovány v jednotlivých clusterech. Tento problém je ovšem vyřešen v programu classifier.m.

7.5 Klasifikační software classifier.m

Pro účely této bakalářské práce je vytvořena speciální funkce v prostředí MATLAB nazvaná classifier.m (příloha V). Na rozdíl od manuálního návrhu neuronové sítě popsaného v podkapitole 7.2 provádí celý zmíněný proces automaticky. Kromě grafických výstupů však poskytuje také výstupy numerické, které lze následně využít pro statistickou analýzu klasifikovaných dat. Jedná se o funkci s variabilním počtem vstupních argumentů, kterou lze použít přímo pomocí příkazového řádku. Vstupními argumenty je určena topologie SOM (součin dimenze 1 a dimenze 2 určuje počet clusterů, do kterých budou vstupní data klasifikována), a dále matice vstupních dat. Uživatel má dále možnost si zvolit, zda mají být zobrazeny grafické výstupy z neuronové sítě (podkapitoly 7.4.1 až 7.4.4) a zda mají být obsahy clusterů uloženy do souboru. Poslední dva vstupní argumenty jsou ovšem volitelné a nemusí být použity. Výstupem z funkce classifier.m je matice „clusters“ obsahující informace o tom, které vzorky vstupních dat náleží daným clusterům. Druhým výstupem z funkce je proměnná „net“, ve které jsou obsažena veškerá nastavení natrénované neuronové sítě.

7.5.1 Popis funkce classifier.m

Prvním řádkem zdrojového kódu (viz příloha V) je předpis funkce classifier.m obsahující všechny vstupní i výstupní argumenty (ne všechny vstupní ovšem musí být použity). Následuje nápověda k funkci v angličtině, kterou lze vyvolat použitím příkazu „help classifier“ v příkazovém řádku v prostředí MATLAB. Uživatel se dozví významy jednotlivých vstupních argumentů a výstupních proměnných.

Následující programový blok podmínkou ošetřuje, zda byly vstupní argumenty zadány ve správném formátu a přiřazuje defaultní hodnoty vstupním argumentům, které nebyly použity.

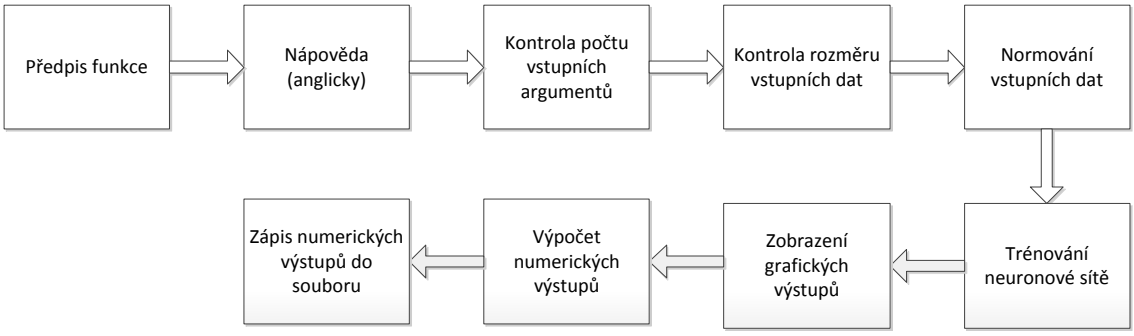
Dále je kontrolován rozměr matice vstupních dat, aby nedošlo k jejich chybnému importu na vstup neuronové sítě. V případě, že jsou počty vzorků a tříd vstupních dat v nesprávném formátu, je matice vstupů transponována. Vstupní data mohou být rozdělena od 2 do nejvýše do 8 tříd. Vzorky vstupních dat (například pacienty) je tedy možné klasifikovat na základě až osmi různých parametrů.

V dalším kroku jsou vstupní data po jednotlivých třídách normována v intervalu 0 až 1. Zde je použita funkce clust_normalize.m popsaná v podkapitole 7.3. Po dokončení normování vstupů je vytvořena výsledná matice, která je posléze importována na vstup vytvořené neuronové sítě.

Průběh trénování neuronové sítě je možné sledovat v GUI nazvaném „nntraintool“. Pokud uživatel v předpisu funkce zvolil možnost zobrazení grafických výstupů, jsou vykresleny grafy popsané v podkapitolách 7.4.1 až 7.4.4.

Po výpočtu výstupní matice z neuronové sítě je zobrazen barevný trojrozměrný graf, který zpřehledňuje graf pozic vah popsaný v podkapitole 7.4.4. Detailní popis trojrozměrného grafu je obsahem následující podkapitoly.

V závěrečné části programu classifier.m jsou výstupní data upravena tak, aby uživatel věděl, které konkrétní vzorky vstupních dat jsou obsaženy v jednotlivých clusterech. Zpětně tak lze analyzovat vstupní data na základě výsledků získaných z klasifikátoru. Zvolil-li uživatel možnost ukládání výstupů do souboru, je v adresáři obsahujícím funkci classifier.m vytvořen soubor neurons.txt.

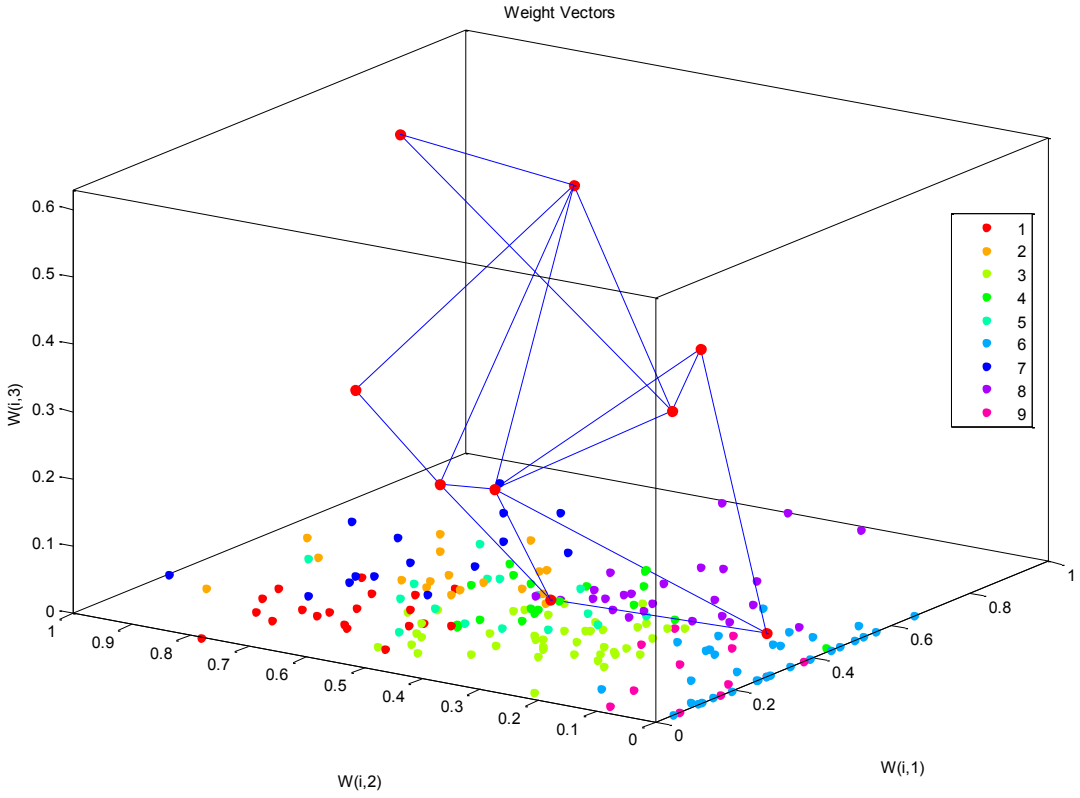


Obr. 7.2 Diagram programových bloků funkce classifier.m

7.5.2 Trojrozměrný graf výstupů neuronové sítě

Vytvořená funkce classifier.m poskytuje možnost zobrazení barevného trojrozměrného grafu, který je vylepšením standardního grafického výstupu SOM Weight Positions (podkapitola 7.4.4) v Neural Network Toolboxu. K tomu je použita funkce „gscatter“. Graf barevně rozlišuje

vzorky vstupních dat náležící daným clusterům tak, aby byla usnadněna grafická analýza výsledků získaných z klasifikátoru. Součástí grafu je také legenda obsahující informaci o tom, který cluster je označen jakou barvou. Klasifikovaná data jsou zobrazena v ploše. Třetí rozměr určuje vzdálenosti neuronů v trojrozměrném prostoru, protože ve dvojrozměrném zobrazení (SOM Weight Positions) vícerozměrného prostoru jsou vazby mezi jednotlivými neurony překrývány a grafická analýza je tedy nemožná.



Obr. 7.3 Trojrozměrný grafický výstup z funkce classifier.m

7.5.3 Numerický výstup z neuronové sítě

Jednou z výstupních proměnných funkce classifier.m je matice pojmenovaná „clusters“. Počet řádků této matice je určen počtem clusterů (neuronů), do kterých jsou vstupní data klasifikována. Počet sloupců je určen počtem prvků v nejobsáhlejším clusteru.

X-tý řádek výstupní matice „clusters“ reprezentuje x-tý cluster. Pokud tento cluster obsahuje méně prvků než nejobsáhlejší cluster, jsou poslední sloupce v x-tém řádku doplněny nulami.

Srovnáme-li grafické zobrazení SOM Sample Hits (podkapitola 7.4.2) s numerickým výstupem (matice „clusters“), počet nenulových prvků v x-tém řádku matice se shoduje s počtem vzorků x-tého neuronu.

8. Kapitola

Testování pro klasifikaci a zhodnocení výsledků

Poslední část této bakalářské práce je věnována testování naprogramovaného klasifikačního software v prostředí MATLAB s názvem classifier.m. Pro účely testování je vytvořen skript nazvaný bc_6_neural_tester.m.

8.1 Vstupní data testu

Vstupní data určená pro test pocházejí z tabulky „tri_mereni“, která obsahuje údaje a metabolické parametry těch pacientů, kteří podstoupili klinické vyšetření 3x v jednom dni (podkapitola 4.3). Klasifikační software vyhodnocuje a zpracovává čtyřrozměrná vstupní data (čtyři metabolické pacientské parametry). Těmito parametry jsou:

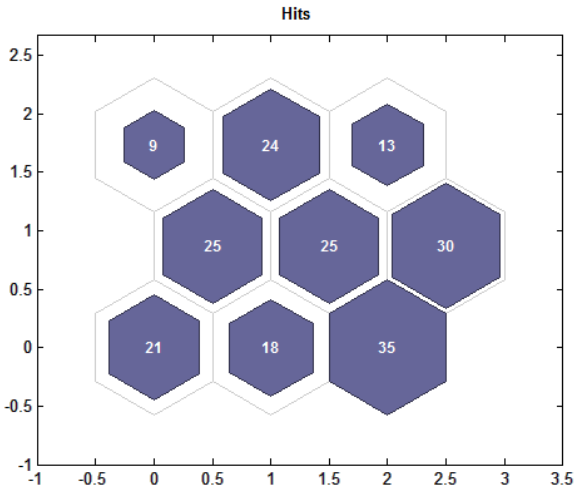
- BMI pacientů
- Procentuální využití tukových energetických substrátů v prvním měření („fat_of_total_1“)
- Procentuální využití tukových energetických substrátů ve druhém měření („fat_of_total_2“)
- Procentuální využití tukových energetických substrátů ve třetím měření („fat_of_total_3“)

Zmíněná data jsou pro účely testu vybrána z toho důvodu, že je možné sledovat vliv podané glukózy na využití energetických substrátů (tuků) u klasifikovaných pacientů. Průběh klinického vyšetření je popsán v podkapitole 2.1. Klasifikovaných je prvních 200 pacientů z tabulky „tri_mereni“. Matice vstupních dat má tedy rozměry 200x4 (200 vzorků ve 4 třídách). Topologie neuronové sítě použité v testu je specifikována rozměry 3x3, pacienti tedy budou rozděleni do 9 clusterů (shluků dat s podobnými parametry).

8.2 Grafická analýza výstupů z neuronové sítě

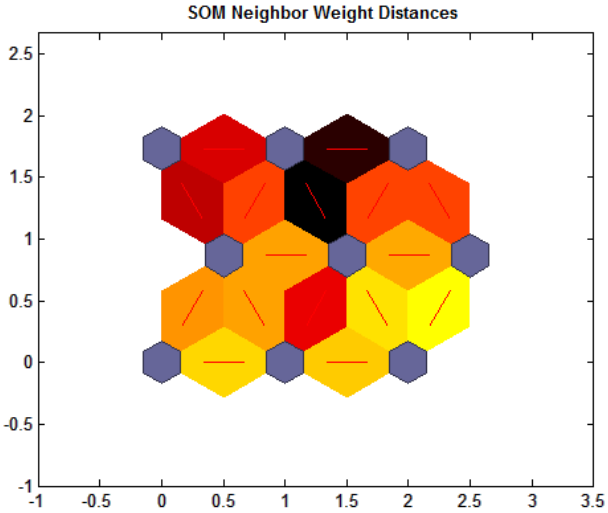
Program classifier.m poměrně rovnoměrně rozdělí pacienty na základě podobností do zmíněných devíti clusterů. Ve všech následujících grafických výstupech začíná číslování neuronů v levém dolním rohu (1. neuron – cluster) a končí v pravém horním rohu (9. neuron).

Nejobsáhlejší je třetí cluster, obsahuje 35 vzorků (pacientů). Naopak nejméně obsáhlý je cluster sedmý, do něhož neuronová síť umístila 9 pacientů. Následující graf (podrobnosti v podkapitole 7.4.2) přehledně zobrazuje počty pacientů v jednotlivých clusterech.



Obr. 8.1 Počty pacientů v jednotlivých clusterech

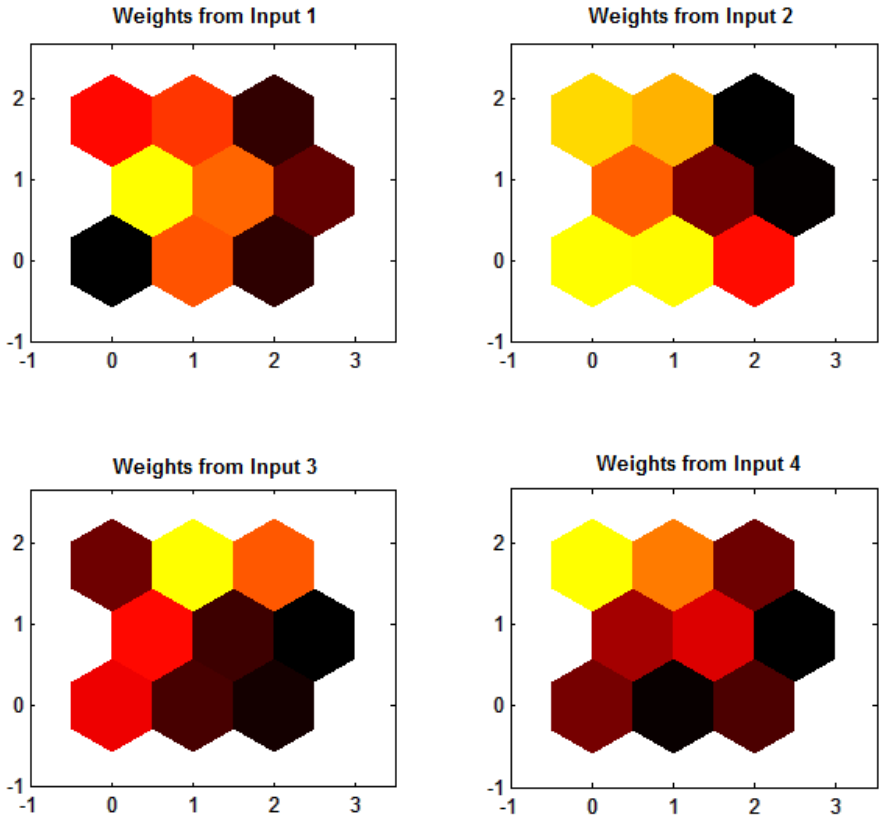
Vzájemné podobností jednotlivých clusterů je možné sledovat na obr. 8.2 (více v podkapitole 7.4.1). Z grafu je patrné, že nejvíce vzájemných podobností mají clustery 3, 5 a 6 (vazby mezi nimi jsou nejsvětlejší). Podobnostmi se vyznačují také clustery 1 a 2. Naopak nejvíce vzájemně odlišné jsou clustery 5 a 8 (vazba je velmi tmavá). Stejně tak dvojice clusterů 8 a 9 má odlišné parametry. Veškerá tvrzení uvedená v tomto odstavci jsou následně ověřena v numerické analýze, která je obsahem následující podkapitoly.



Obr. 8.2 Vzájemné podobnosti reprezentované vazbami mezi clustery

Posledním analyzovaným grafickým výstupem je zobrazení vah jednotlivých vstupů (podkapitola 7.4.3). Vzhledem k tomu, že vstupní data jsou čtyřrozměrná, obsahuje zmíněné zobrazení čtyři grafy. Každý z těchto grafů reprezentuje vliv daného vstupního parametru na nastavení synapsí (vah) mezi neurony, které má za následek shlukování vstupních dat

do clusterů. Zároveň je možné srovnáním barev vybraných neuronů zkoumat vzájemné závislosti vstupních parametrů. Grafy reprezentují vstupní parametry v tomtéž pořadí, které je uvedeno v podkapitole 8.1. Levý horní graf tedy reprezentuje vliv parametru BMI na nastavení vstupních vah neuronů, pravý horní graf procentuální využití tukových energetických substrátů v prvním měření atd.



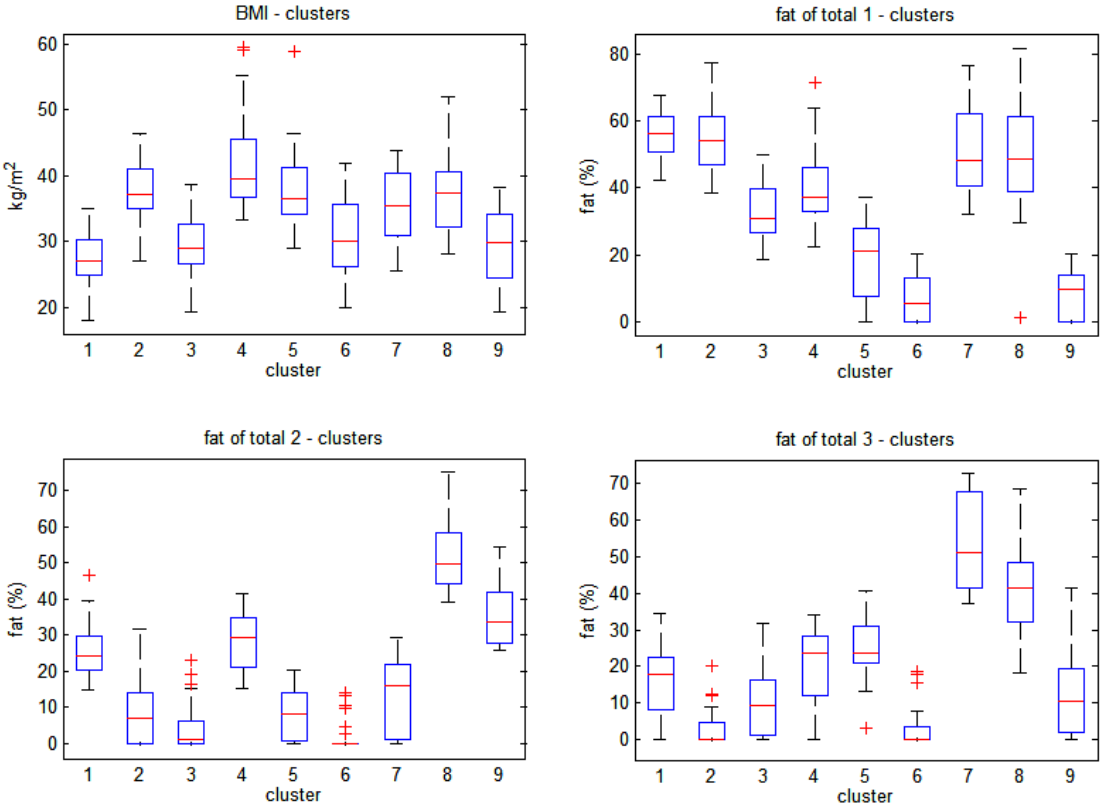
Obr. 8.3 Zobrazení vah jednotlivých vstupů

Ze všech grafů je patrné, že 6. cluster obsahuje vzorky (pacientská data), která mají jen malý vliv na nastavení vah neuronů. Statisticky to znamená, že zmíněné metabolické parametry nabývají u pacientů v 6. clusteru zanedbatelných hodnot. Podobné tvrzení lze prokázat i u 9. clusteru. Naopak pacienti klasifikovaní v 8. clusteru mají významné metabolické hodnoty, osmý neuron má vždy světlou barvu. Numerická vyjádření statistických hodnot jsou obsahem následující podkapitoly. Trojrozměrný grafický výstup reprezentující rozdělení clusterů a nastavení a zobrazující reálné vzdálenosti mezi neurony je součástí přílohy III.

Grafická analýza výstupů z klasifikátoru je vhodná k získání základního přehledu o analyzovaných a klasifikovaných datech. Všechna tvrzení je ovšem nutné potvrdit numerickou analýzou, která je v technické praxi nezbytnou součástí každé výzkumné činnosti.

8.3 Numerická analýza výstupů z neuronové sítě

Klasifikovaná data jsou statisticky vyhodnocena ve skriptu bc_6_neural_tester.m. K přehledné reprezentaci statistických výsledků je použit krabicový graf, jehož popis je uveden v kapitole 5.3. Postup numerické analýzy je takový, že z matice „clusters“, která je jedním z datových výstupů programu classifier.m, jsou získány informace o tom, kteří konkrétní pacienti jsou klasifikováni v jednotlivých clusterech. Následně je provedeno statistické vyhodnocení všech vybraných metabolických parametrů pacientů z matice vstupních dat a výsledky jsou zobrazeny po parametrech do krabicových grafů.



Obr. 8.4 Krabicové grafy metabolických parametrů pacientů klasifikovaných do clusterů

Při pohledu na grafy je zjevné, že klasifikační software rozlišil pacienty s podobnými metabolickými parametry a zařadil je do clusterů, které se menší či větší mírou vzájemně liší. Srovnáním grafů „fat of total“ 1 až 3 lze sledovat vliv podané glukózy během měření na procentuální využití tukových energetických substrátů. Pacienti v 1. clusteru před podáním glukózy (1. měření – graf „fat of total 1“) spalují tuk velmi dobře, medián těchto hodnot je roven 56,1%. Po podání glukózy však tito pacienti využívají tukový energetický substrát podstatně hůř, ve druhém měření má medián hodnotu 24,4% (graf „fat of total 2“) a v měření třetím pouze 17,8%. Opakem pacientů v 1. clusteru jsou pacienti klasifikovaní v 8. clusteru. Na

počátku měření je medián využití tuku roven hodnotě 48,5%. V měření druhém dokonce hodnota mediánu vzroste na 49,7% a v měření třetím klesne na 41,45%.

Tabulka 8.1 Hodnoty mediánu metabolických parametrů v clusterech

		1. cluster	2. cluster	3. cluster	4. cluster	5. cluster	6. cluster	7. cluster	8. cluster	9. cluster
	Počet pacientů	21	18	35	25	25	30	9	24	13
BMI	Medián (%)	27,005	37,192	29,044	39,519	36,612	30,118	35,511	37,461	29,761
FAT 1		56,100	54,250	31,000	37,300	20,900	5,550	48,000	48,500	9,500
FAT 2		24,400	6,850	1,200	29,200	8,000	0,000	16,100	49,700	33,500
FAT 3		17,800	0,000	9,400	23,500	23,800	0,000	51,100	41,450	10,600

Při pohledu na metabolické hodnoty 6. clusteru je patrné, jak navzájem korespondují grafické výstupy neuronové sítě a výstupy numerické analýzy. Na 2., 3. a 4. grafu v obr. 8.3 je šestý cluster vyznačen černou barvou, což znamená, že statistické parametry jsou z hlediska nastavení vah mezi klasifikujícími neurony nevýznamné (viz podkapitola 8.2). Při pohledu na obr. 8.4 je vidět, že medián hodnot spalování tuků pacientů v 6. clusteru je v prvním měření roven pouze 5,5%, v ostatních měření je medián roven 0%. Znamená to, že klasifikační software zařadil do tohoto clusteru pacienty, kteří po podání glukózy spalují tuk velmi špatně nebo pacienty, jejichž data z měření tohoto metabolického parametru nejsou dostupná (jsou rovna nule). V 6. clusteru je klasifikováno 30 pacientů, viz obr. 8.1.

Vzájemné odlišnosti clusterů 5, 8 a 9 jsou evidentní při srovnání obr. 8.2 a obr. 8.4. Na grafickém výstupu na obr. 8.2 jsou vazby mezi těmito neurony (clustery) velmi tmavé, což reprezentuje velkou míru odlišnosti. V grafech „fat of total“ 1 až 3 na obr. 8.4 je zřetelné, že se hodnoty mezikvartilových intervalů u clusterů 5, 8 a 9 téměř nepřekrývají. Tvrzení lze demonstrovat například na třetím měření využití tukových energetických substrátů clusterů 5, 8 a 9 (čtvrtý graf na obr. 8.4). Horní kvartil 9. clusteru má hodnotu 19,23%, dolní kvartil 5. clusteru 21,00%, jeho horní kvartil 31,05% a konečně dolní kvartil 8. clusteru 32,25%.

Při numerické analýze výstupů z klasifikátoru je nutné vzít v potaz, že některé vzorky vstupních dat jsou tzv. odlehlými měřeními, v grafech na obr. 8.4 vyznačenými červenými křížky. S nimi si ovšem neuronová síť při klasifikaci poradí, příkladem toho je výše zmíněný 6. cluster.

Z obr. 8.4 vyplývá skutečnost, že hodnota BMI má na nastavení vah mezi neurony menší vliv než hodnoty využití tukových energetických substrátů, jelikož clustery parametru BMI mají rovnoměrnější statistické parametry a jsou vzájemně více podobné.

8.4 Data z testu

Aby bylo možné celý výše popsaný test programu opakovat, jsou výstupy uloženy v souboru experiment.mat, který je součástí příloh k této bakalářské práci. Soubor obsahuje matici vstupních dat, matici výstupů a specifikaci neuronové sítě. Součástí přílohy IV této práce je tabulka obsahující veškerá statistická data získaná během testování klasifikátoru.

Závěr

Ve své bakalářské práci jsem se zabýval možnostmi analýzy, zpracování a klasifikace velkého množství naměřených metabolických pacientských dat na základě moderních metod umělé inteligence. Po studiu průběhu klinických vyšetření pacientů nemocnice v Hradci Králové, ze kterých jsem naměřená data získal, bylo nutné navrhnout a vytvořit rozsáhlou databázi pro jejich ukládání. Pro tento účel jsem použil databázový systém MySQL. Kombinace systému MySQL a programovacího prostředí MATLAB, ve kterém jsem všechny skripty této bakalářské práce vytvářel, se osvědčila. Vzájemná komunikace mezi těmito prostředími je velmi intuitivní a v průběhu mé práce jsem s ní neměl žádné problémy.

V první fázi realizace této bakalářské práce jsem se věnoval návrhu algoritmů, které jsou schopny z textových souborů, ve kterých jsou jednotlivá měření pacientů uložena, získat potřebné metabolické parametry a údaje o pacientech a uložit je posléze do databáze. Tvorba databáze mi zabrala mnoho času, jelikož jsem vždy po úspěšném vyřešení dílčích problémů narazil na další komplikace. Jednalo se především o problémy se soubory, které byly neúplné či poškozené, a bylo tedy nutné je filtrovat, aby negativně neovlivňovaly další průběh zpracování dat.

Po dokončení výsledné databáze bylo možné přistoupit k jednoduché statistické analýze z důvodu optimalizace klasifikačních algoritmů a správného výběru testovacích vstupních dat. Zjistil jsem, že drtivá většina pacientů trpí určitým stupněm obezity. V důsledku této skutečnosti jsem po dokončení klasifikačního programu zvolil testovací vstupní data.

Stěžejní částí mé bakalářské práce bylo studium možností neuronových sítí s architekturou self-organizing map v prostředí MATLAB a následný návrh a realizace klasifikačního programu. Jádro programu je tvořeno algoritmy obsaženými v Neural Network Toolboxu. Jedná se o program s variabilním počtem vstupních argumentů, ve kterém jsou ošetřena vstupní data zadaná ve špatném formátu a následně normována pro potřeby neuronové sítě. Aplikace generuje grafické i numerické výsledky, na základě kterých lze interpretovat klasifikovaná data. Význam jednotlivých výstupů je součástí příslušných kapitol této práce. Velkou výhodou programu je skutečnost, že se jedná o obecný algoritmus, který je možné použít nejen ke klasifikaci komplexních metabolických pacientských dat, ale v podstatě ke klasifikaci kterýchkoliv vícerozměrných vstupů. Program je schopen klasifikovat data na základě až osmi parametrů. Jeho nevýhodou je fakt, že z důvodu náhodné inicializace hodnot vah mezi jednotlivými neurony modelu neuronové sítě dochází při každém použití programu k trochu odlišným výsledkům, což může být nevyhovující při použití v determinovaných systémech.

V závěru práce jsem se věnoval testování vytvořeného software. Jako testovací parametry vstupních dat jsem použil stupeň obezity pacientů (BMI) a procentuální využití tuku ve všech třech měřeních. Program klasifikoval pacienty na základě těchto parametrů do různorodých

skupin (clusterů), což je zřejmé především ze statistické analýzy jednotlivých parametrů pacientů. Statistické výstupy z klasifikátoru jsou popsány v kapitole 8, v příloze IV je pak zařazena tabulka statistické analýzy.

Co se týče dalšího vývoje mé bakalářské práce, je nyní možné přistoupit k lékařské interpretaci získaných výsledků a studovat komplexní vztahy mezi metabolickými parametry, neboť program je silným nástrojem ke klasifikaci dat na základě mnoha parametrů. Pan prof. Martiník z nemocnice v Hradci Králové, který mi poskytl veškerá pacientská data, má nyní možnost se do hloubky zabývat analýzou výsledků a pomoci tím vývoji efektivnější léčby pacientů. V případě zájmu ze strany nemocnice v Hradci Králové jsem ochoten rozšířit možnosti klasifikačního programu.

Na závěr bych chtěl dodat, že jsem splnil všechny body stanovené v zadání mé bakalářské práce, důkazem tohoto tvrzení je funkční databáze v systému MySQL a program classifier.m.

Literatura a použité zdroje

[1] MARTINÍK, K. *Obezita, nadváha: Od teorie k praxi.* 1. vydání. Hradec Králové: Garamon s.r.o., 2008. 151 s. ISBN 978-80-86472-37-9.

[2] *Deltatrac II: Weltwait der genaueste und vielseitigste Metabolic Monitor.* Finland: Datex-Ohmeda Division, 2003. 115 s.

[3] *Hoyer* [online]. 2005 [cit. 2011-01-09]. Deltatrac II. Dostupné z WWW: <http://www.hoyer.cz/produkty/produkty-minulych-instalaci/intenzivni-pece/deltatrac-ii/>

[4] *Wikipedie* [online]. 2011 [cit. 2011-01-14]. Index tělesné hmotnosti. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Index_t%C4%9Blesn%C3%A9_hmotnosti>

[5] *Mysql* [online]. 2010 [cit. 2011-03-08]. Why MySQL. Dostupné z WWW: <http://www.mysql.com/why-mysql/>

[6] *Mathworks* [online]. 2009 [cit. 2010-10-15]. Connect to database – MATLAB. Dostupné z WWW: <http://www.mathworks.com/help/toolbox/database/ug/database.html>.

[7] CVRČEK, Pavel. *Zive.cz* [online]. 2001 [cit. 2010-12-19]. Začínáme s MySQL 11 – import a export dat. Dostupné z WWW: <http://www.zive.cz/clanky/zaciname-s-mysql-11--import-a-export-dat/sc-3-a-103982/default.aspx>.

[8] *Amper.ped.muni.cz* [online]. 1997 [cit. 2010-11-25]. Popisné statistiky. Dostupné z WWW: <http://amper.ped.muni.cz/jenik/nejistoty/html_tree/node13.html>

[9] *Wood.mendelu.cz* [online]. 2009 [cit. 2010-11-25]. Typy grafů v R. Dostupné z WWW: <http://wood.mendelu.cz/cz/sections/FEM/?q=node/82>

[10] ABONYI, Janos. *Mathworks* [online]. 2005 [cit. 2011-03-22]. Clustering toolbox. Dostupné z WWW: <http://www.mathworks.com/matlabcentral/fileexchange/7486>

[11] POKORNÝ, M., PETRÁNEK, P. *Systémy s umělou inteligencí.* Ostrava, 2005. 121 s. Studijní skripta. VŠB – Technická univerzita Ostrava, Fakulta elektrotechniky a informatiky.

[12] JIRSÍK, V., HRÁČEK, P. *Neuronové sítě, expertní systémy a rozpoznávání řeči.* Brno. 106 s. Studijní skripta. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií.

[13] TUČKOVÁ, J. *Vybrané aplikace umělých neuronových sítí při zpracování signálů.* Praha: Česká technika – nakladatelství ČVUT, 2009. 224 s. ISBN 978-80-01-04229-8

Seznam příloh

Přílohy

Příloha I: Příklad datového výstupu z přístroje Deltatrac II

PATIENT:

SEX	female
DATE OF BIRTH	05-Mar-74
AGE	35 yr
HEIGHT	160 cm
WEIGHT	105 kg
NITROGEN EXCRETION	13 g/24h
BODY SURFACE AREA	2.06 m2
BASAL METABOLIC RATE	1790 kcal/24h (H-B)
	1710 kcal/24h (Fleisch)

CANOPY MODE ADULT
RESULTS IN STPD
NO ARTIFACT SUPPRESSION
GAS TEMPERATURE 32.0 C
AMBIENT PRESSURE 739 mmHg

MEASUREMENT STARTED	09:57 06-Jan-2010
TILL	10:09 06-Jan-2010
DURATION	00:11
ARTIFACTS AND INTERRUPTS	00:00

CALIBRATIONS:	AMB. CO2	CO2 MEAS./SET	O2 MEAS./SET
14-Jul-2009 14:24	0.06 %	5.00 / 5.00 %	95.1 / 95.0 %

FLOW SETTING: ADULT 44.3 l/min

MEASUREMENT RESULTS:

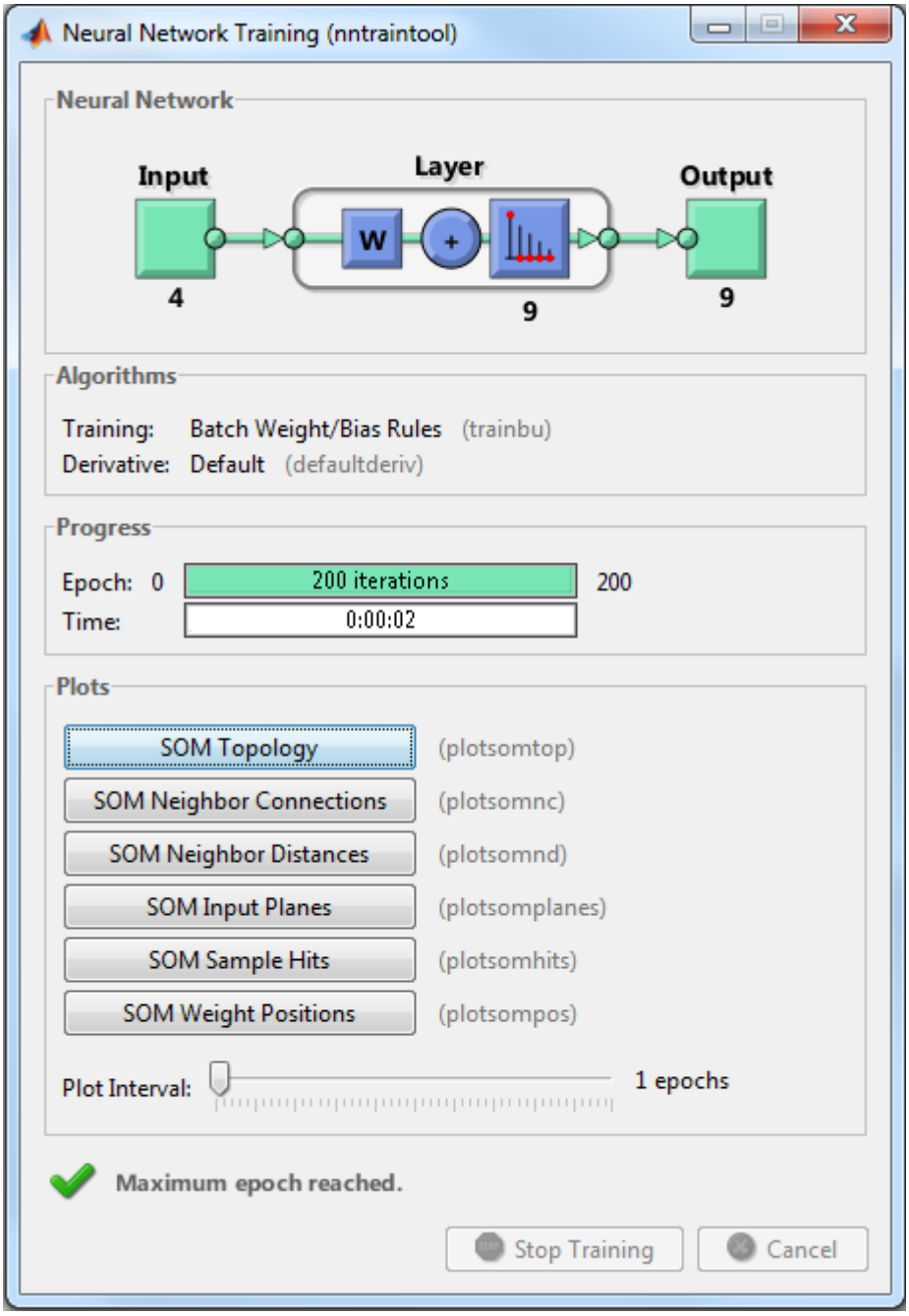
	VCO2 ml/min	VO2 ml/min	RQ	EE kcal/24h
MEAN	286	322	0.89	2250
STANDARD	36	46	0.07	310
DEVIATION	12.6 %	14.2 %	7.6 %	13.7 %
(N = 11)				

DIFFERENCE FROM BASAL METABOLIC RATE: + 26 % (H-B)
NON-PROTEIN RQ 0.90

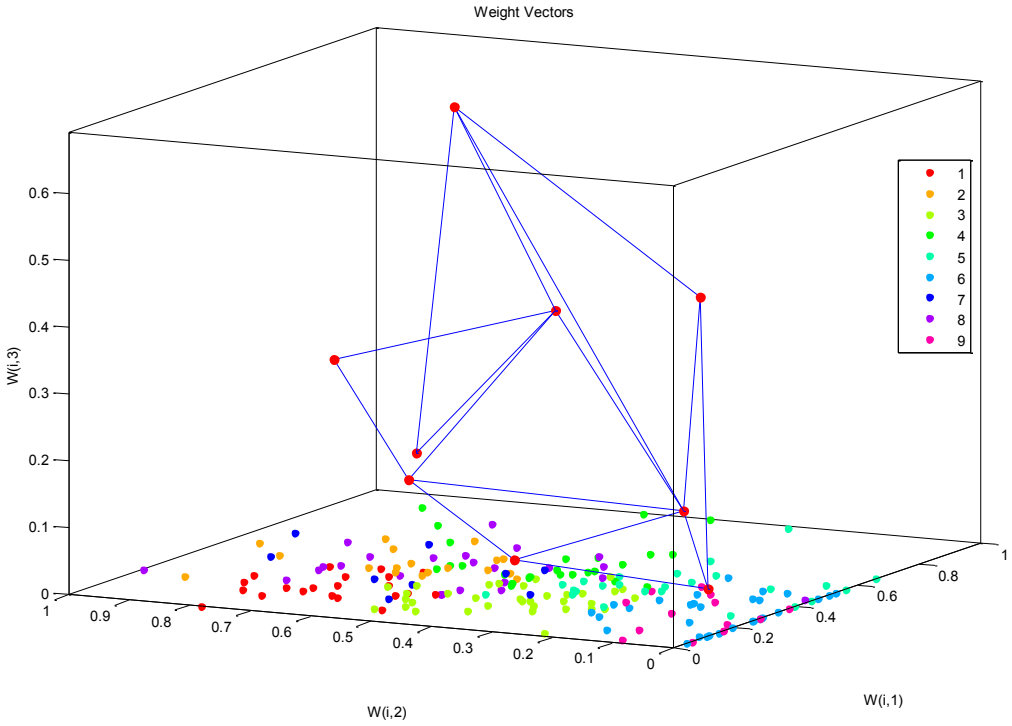
ENERGY SUBSTRATE UTILIZATIONS:

	g/24h	% OF TOTAL
CARBOHYDRATE	311	57.9
FAT	63	26.5
PROTEIN	81	15.6

End of Report
Pas: 107
Boky: 125
Rcislo: xxxxxxxxxx



Příloha II: Rozhraní trénování neuronové sítě



Příloha III: Trojrozměrný grafický výstup z klasifikátoru

Příloha IV: Tabulka všech statistických parametrů clusterů získaných v rámci testování programu classirier.m

		1. cluster	2. cluster	3. cluster	4. cluster	5. cluster	6. cluster	7. cluster	8. cluster	9. cluster
	Počet pacientů	21	18	35	25	25	30	9	24	13
BMI	Median	27,005	37,192	29,044	39,519	36,612	30,118	35,511	37,461	29,761
	Horní kvartil	30,258	40,957	32,609	45,629	41,333	35,749	40,285	40,684	34,115
	Dolní kvartil	24,997	35,056	26,602	36,750	34,214	26,107	30,937	32,153	24,418
	Mezikvartilové rozpětí	5,261	5,901	6,007	8,879	7,119	9,642	9,348	8,531	9,697
	Horní přesah	35,013	46,295	38,725	55,249	46,309	41,778	43,858	51,957	38,296
	Dolní přesah	18,066	26,989	19,265	33,387	29,069	19,921	25,649	28,086	19,313
FAT 1	Median	56,100	54,250	31,000	37,300	20,900	5,550	48,000	48,500	9,500
	Horní kvartil	61,150	61,100	39,650	46,075	27,825	13,000	62,125	61,400	13,800
	Dolní kvartil	50,550	46,800	26,550	32,925	7,400	0,000	40,700	38,700	0,000
	Mezikvartilové rozpětí	10,600	14,300	13,100	13,150	20,425	13,000	21,425	22,700	13,800
	Horní přesah	67,700	77,200	49,900	63,900	37,000	20,200	76,600	81,500	20,200
	Dolní přesah	42,400	38,500	18,600	22,300	0,000	0,000	32,200	29,600	0,000
FAT 2	Median	24,400	6,850	1,200	29,200	8,000	0,000	16,100	49,700	33,500
	Horní kvartil	29,850	14,200	6,100	34,700	14,200	0,000	21,750	58,350	42,000
	Dolní kvartil	20,425	0,000	0,000	21,300	0,900	0,000	1,200	44,100	27,850
	Mezikvartilové rozpětí	9,425	14,200	6,100	13,400	13,300	0,000	20,550	14,250	14,150
	Horní přesah	39,500	31,800	15,200	41,400	20,300	0,000	29,400	75,200	54,300
	Dolní přesah	15,000	0,000	0,000	15,400	0,000	0,000	0,000	39,300	25,700
FAT 3	Median	17,800	0,000	9,400	23,500	23,800	0,000	51,100	41,450	10,600
	Horní kvartil	22,350	4,800	16,150	28,375	31,050	3,500	67,700	48,500	19,225
	Dolní kvartil	8,075	0,000	1,200	12,225	21,000	0,000	41,400	32,250	1,875
	Mezikvartilové rozpětí	14,275	4,800	14,950	16,150	10,050	3,500	26,300	16,250	17,350
	Horní přesah	34,400	9,000	31,600	34,200	40,700	7,800	72,600	68,400	41,300
	Dolní přesah	0,000	0,000	0,000	0,000	13,300	0,000	37,000	18,200	0,000

Příloha V: Zdrojový kód programu classifier.m

```
function [clusters,net] = classifier(dimension1, dimension2, input,
show_plots, save_clusters)
    %CLASSIFIER Generates cluster analysis matrices and plots using
    %self-organizing map neural network architecture
    %   [CLUSTERS,NET] = CLASSIFIER(DIMENSION1, DIMENSION2, INPUT)
    %   calculates cluster matrices. Input arguments DIMENSION1 and
    %   DIMENSION2 must be real pozitive integers which define size of
    %   self-organizing map. More dimensions, more neurons (clusters).
    %   However, recommended size is 3x3 (9 clusters) or 5x5. (25
clusters).
    %   Variable INPUT is M-by-N  matrix, where M represents number
    %   of samples (e.g. measurements) and N number of elements
(classes).
    %   N must belong to interval <2,8>.
    %   Output variable CLUSTERS contains information which of input
    %   samples belong to the cluster. Variable NET contains parameter
    %   description of trained self-organizing map.
    %   Each row of CLUSTERS represent one neuron (cluster).
    %   [CLUSTERS,NET] = CLASSIFIER(DIMENSION1, DIMENSION2, INPUT,
SHOW_PLOTS)
    %   shows also output plots of self-organizing map, if SHOW_PLOTS
is
    %   'on'.
    %   See also PLOTSOMND, PLOTSOMPLANES, PLOTSOMHITS, PLOTSOMPOS to
find
    %   out what these plots represent.
    %
    %   [CLUSTERS,NET] = CLASSIFIER(DIMENSION1, DIMENSION2, INPUT,
SHOW_PLOTS, SAVE_CLUSTERS)
    %   creates file neurons.txt containing variable clusters if
SAVE_CLUSTERS
    %   is 'on'.



    %Number of input arguments checking
    error(nargchk(3,5,nargin,'struct'));
    %Default input arguments setting
    if (nargin < 4)
        show_plots = 'off';
    end
    if (nargin < 5)
        save_clusters = 'off';
    end

    %Transpose if wrong-shaped input matrix entered
    format = size(input);
    if format(2) > format(1)
        input = input';
    end
    %Check for input matrix dimension
    format = size(input);
    description = format(2);
    %Normalizes input samples - values in range 0-1
```

```
    if description == 1
        msgbox('Input vector must have at least two elements for each
sample!','Cannot calculate!','warn');
    elseif description == 2
        data1.X = input(:,1);
        data2.X = input(:,2);
        data1 = clust_normalize(data1,'range');
        data2 = clust_normalize(data2,'range');
        neural_input = [data1.X data2.X];
    elseif description == 3
        data1.X = input(:,1);
        data2.X = input(:,2);
        data3.X = input(:,3);
        data1 = clust_normalize(data1,'range');
        data2 = clust_normalize(data2,'range');
        data3 = clust_normalize(data3,'range');
        neural_input = [data1.X data2.X data3.X];
    elseif description == 4
        data1.X = input(:,1);
        data2.X = input(:,2);
        data3.X = input(:,3);
        data4.X = input(:,4);
        data1 = clust_normalize(data1,'range');
        data2 = clust_normalize(data2,'range');
        data3 = clust_normalize(data3,'range');
        data4 = clust_normalize(data4,'range');
        neural_input = [data1.X data2.X data3.X data4.X];
    elseif description == 5
        data1.X = input(:,1);
        data2.X = input(:,2);
        data3.X = input(:,3);
        data4.X = input(:,4);
        data5.X = input(:,5);
        data1 = clust_normalize(data1,'range');
        data2 = clust_normalize(data2,'range');
        data3 = clust_normalize(data3,'range');
        data4 = clust_normalize(data4,'range');
        data5 = clust_normalize(data5,'range');
        neural_input = [data1.X data2.X data3.X data4.X data5.X];
    elseif description == 6
        data1.X = input(:,1);
        data2.X = input(:,2);
        data3.X = input(:,3);
        data4.X = input(:,4);
        data5.X = input(:,5);
        data6.X = input(:,6);
        data1 = clust_normalize(data1,'range');
        data2 = clust_normalize(data2,'range');
        data3 = clust_normalize(data3,'range');
        data4 = clust_normalize(data4,'range');
        data5 = clust_normalize(data5,'range');
        data6 = clust_normalize(data6,'range');
        neural_input = [data1.X data2.X data3.X data4.X data5.X
data6.X];
    elseif description == 7
        data1.X = input(:,1);
        data2.X = input(:,2);
```

```
        data3.X = input(:,3);
        data4.X = input(:,4);
        data5.X = input(:,5);
        data6.X = input(:,6);
        data7.X = input(:,7);
        data1 = clust_normalize(data1,'range');
        data2 = clust_normalize(data2,'range');
        data3 = clust_normalize(data3,'range');
        data4 = clust_normalize(data4,'range');
        data5 = clust_normalize(data5,'range');
        data6 = clust_normalize(data6,'range');
        data7 = clust_normalize(data7,'range');
        neural_input = [data1.X data2.X data3.X data4.X data5.X
data6.X data7.X];
    elseif description == 8
        data1.X = input(:,1);
        data2.X = input(:,2);
        data3.X = input(:,3);
        data4.X = input(:,4);
        data5.X = input(:,5);
        data6.X = input(:,6);
        data7.X = input(:,7);
        data8.X = input(:,8);
        data1 = clust_normalize(data1,'range');
        data2 = clust_normalize(data2,'range');
        data3 = clust_normalize(data3,'range');
        data4 = clust_normalize(data4,'range');
        data5 = clust_normalize(data5,'range');
        data6 = clust_normalize(data6,'range');
        data7 = clust_normalize(data7,'range');
        data8 = clust_normalize(data8,'range');
        neural_input = [data1.X data2.X data3.X data4.X data5.X
data6.X data7.X data8.X];
    else
        msgbox('Cannot calculate!','Input class dimension is to
high!!','error');
    end
    neural_input = neural_input';

    %Creates self-organizing map neural network architecture
    net = selforgmap([dimension1, dimension2]);
    net = train(net, neural_input);
    %Shows neural network training GUI
    nntraintool;

    %Shows neural network output plots
    if strcmp(show_plots,'on');
        figure;
        plotsomnd(net,neural_input);
        figure;
        plotsomplanes(net,neural_input);
        figure;
        plotsomhits(net,neural_input);
        figure;
        plotsompos(net,neural_input);
    end
```

```
    %Calculates item positions and shows colourful 3D plot
    output = net(neural_input);
    item_position = vec2ind(output);
    figure;
    gscatter(neural_input(1,:),neural_input(2,:),item_position);
    hold on;
    plotsom(net.iw{1,1},net.layers{1}.distances);
    hold off;
    axis([0 1 0 1]);

    %Generates clusters matrix - each row represents one neuron -
cluster
    for i = 1:dimension1*dimension2
        row = find(item_position == i);
        extent = size(row);
        clusters(i,1:extent(2)) = row;
    end

    %Save-to-file cycle - saves data from workspace using diary
    if strcmp(save_clusters,'on');
        delete('neurons.txt');
        diary neurons.txt;
        for j = 1:dimension1*dimension2
            elements = find(clusters(j,:) ~=0);
            cluster_No = clusters(j,1:length(elements));

            fprintf('%d. cluster contains these elements:\n', j);
            cluster_No
        end
        fprintf('\n\n\n\nSummarization:\n');
        clusters
        fprintf('Each row represents one neuron - cluster.\n');
        diary off;
    end
end
```